Structuration de scènes architecturales complexes en vue de simulation d'éclairage

Daniel Meneveaux, Eric Maisel, Florent Coudret, Kadi Bouatouch

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Structuration de scènes architecturales complexes en vue de simulation d'éclairage.

D. Meneveaux, E. Maisel, F. Coudret, K. Bouatouch

N° 2981

Septembre 1996

THÈME 3

ISSN 0249-6399

Structuration de scènes architecturales complexes en vue de simulation d'éclairage.

D. Meneveaux, E. Maisel, F. Coudret, K. Bouatouch

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Siames

Rapport de recherche n° 2981 — Septembre 1996 — 65 pages

Résumé : Ce rapport décrit une méthode permettant d'effectuer une simulation d'éclairage dans des environnements architecturaux complexes pouvant contenir des milliers voire des millions de primitives géométriques et nécessitant une taille mémoire considérable. L'approche utilisée est basée sur une subdivision automatique de ces scènes en sous-scènes appelées *cellules* devant respecter au mieux la topologie de l'environnement. La subdivision s'opère à l'aide des plans contenant les objets (facettes planes) les plus occlusifs. Dans notre approche, nous faisons l'hypothèse que les objets les plus occlusifs sont axiaux (i.e. leur normale est parallèle à l'un des trois axes du repère de la scène). Le résultat de cette subdivision est un arbre binaire dont les feuilles sont des cellules. Une fois les cellules obtenues, un graphe determinant les relations de visibilité inter-cellules est établi. Ensuite, nous proposons une analyse et une interprétation des différents résultats obtenus grâce à notre méthode.

Mots-clé : Synthèse d'image, environnements architecturaux, complexité, simulation d'éclairage, visibilité.

(Abstract: pto)

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

Partitioning complex architectural environments for lighting simulation.

Abstract:

This report describes a method making possible lighting simulation within complex architectural environments composed of thousands or millions of primitive objects and requiring important memory resources. Our approach consists in automatically partitioning such scenes into *cells* fitting as much as possible the topology of the scene. This partitioning is performed with the help of the planes containing the most occluding objects (polygons). Our method assumes that these planes are supposed to be axials (i.e. their normal is aligned with one axis of the world coordinate system). The partitioning process results in a binary tree whose leaves are the cells. Once these latter have been determined, a graph expressing visibility relationships between cells is built. Then some results provided by our method are analysed and interpreted.

Key-words: Image synthesis, architectural environments, complexity, lighting simulation, visibility.

1 Introduction

Un des objectifs de notre projet concerne la possibilité d'explorer en temps réel des environnements complexes visualisés avec un rendu photo-réaliste. Pour atteindre le photo-réalisme en synthèse d'images, il est nécessaire de simuler correctement l'éclairage des scènes visualisées. La méthode de radiosité répond à cette demande en apportant, en outre, une totale indépendance des calculs par rapport au point d'observation choisi dans une scène. Cette propriété intéresse naturellement l'éclairage d'environnements dans lesquels on peut se déplacer rapidement, et en particulier, les environnements architecturaux (intérieurs d'immeubles ou artères d'une ville par exemple). L'algorithme de radiosité exige cependant de "mailler" la surface des objets (c'est-à-dire de la subdiviser en petites surfaces : les carreaux), puis d'établir les relations de visibilité existantes entre les carreaux obtenus. Malheureusement, pour obtenir une bonne précision dans les calculs d'éclairement, un nombre élevé de carreaux est souvent nécessaire, même pour des scènes peu complexes (c'est-à-dire ne contenant qu'un faible nombre d'objets). Cette contrainte augmente considérablement les coûts de calcul et de taille mémoire nécessaire au calcul des échanges d'énergie lumineuse, et limite d'autant la taille des scènes manipulées.

Il est également intéressant de pouvoir traverser et visualiser en temps réel de telles scènes. Le problème principal est d'afficher uniquement les éléments de la scène (par exemple l'ensemble des polygones) effectivement visibles de l'observateur. Pour cela, des calculs de visibilité sont nécessaires pour identifier ces éléments. Malheureusement, la taille des scènes à visualiser est bien souvent trop importante pour pouvoir afficher suffisamment d'images par seconde, diminuant ainsi la fluidité de l'animation. En effet, la limitation vient du nombre de primitives par seconde que les algorithmes de visibilité (même cablés) peuvent traiter.

Pour diminuer ces complexités spatiales et temporelles, ils est possible d'établir certaines informations de visibilité globale sur une scène. Peu de méthodes ont été proposées dans la littérature (celles-ci seront détaillées dans la section 2). Puisque l'on ne peut prendre en compte l'intégralité d'une scène complexe pour calculer, par exemple, rapidement son éclairement, l'idée est de structurer celle-ci en cellules puis d'établir un graphe de visibilité entre ces cellules. Ainsi, l'éclairage d'une cellule ne nécessite plus la contribution de toute la scène mais uniquement celle des autres cellules avec lesquelles elle possède des relations de visibilité. De même, ces relations de visibilité permettent d'accélérer la phase de visualisation d'une scène. Il suffit de limiter l'ensemble des objets potentiellement visibles à ceux présents dans les cel-

lules visibles de la cellule occupée par l'observateur. Ainsi, le nombre de primitives que doivent traiter les algorithmes de visibilité est diminué, ce qui accélère d'autant les temps de traitement et d'affichage d'une image. Dans les deux cas, il devient possible de traiter des scènes plus importantes en réduisant sensiblement les complexités évoquées plus haut.

Notre approche reprend cette direction en étudiant de manière concrète, et souvent plus approfondie, les facteurs essentiels à une bonne structuration de la scène en cellules. Plusieurs cas particuliers, peu souvent évoqués dans la littérature, sont également traités. Nous proposons ainsi une méthode efficace, basée sur une technique de subdivision binaire de l'espace (Binary Space Partitioning), pour découper une scène en cellules, ainsi qu'un algorithme simple et rapide pour déterminer les visibilités des cellules entre elles. Notre contribution concerne plus particulièrement la détermination des critères de subdivision conduisant à une structuration "fine" des scènes, ainsi qu'au traitement de certains cas particuliers. De plus, les résultats de ces travaux ne se limitent pas aux calculs de radiosité d'une scène, mais peuvent également être utilisés pour d'autres algorithmes de rendu tels que le tampon de profondeur (Z-buffer) ou encore le lancer de rayon.

La suite de ce document est organisée de la manière suivante : la section 2 constitue un état de l'art sur les solutions déjà proposées dans la littérature pour résoudre les problèmes de visibilité intervenant dans la visualisation de scènes complexes. L'idée commune des auteurs repose sur une structuration de la scène en cellules et l'utilisation d'un graphe de visibilité. Les différentes méthodes présentées sont discutées et comparées à la fin de cette section.

Notre contribution est présentée à la section 3. Après une brève description du vocabulaire employé et des hypothèses que nous nous imposons, nous nous attachons à décrire notre propre méthode de structuration de la scène en cellules et les critères de découpage retenus. La construction du graphe de visibilité des cellules entre elles est également détaillée: L'algorithme utilisé reste simple et plusieurs améliorations possibles seront signalées. Une description des structures essentielles manipulées (arbre BSP, cellules, graphe de visibilité) termine cette section.

la section 4 présente une étude des résultats permettant d'évaluer notre méthode et les critères employés sur des scènes de complexité différentes. Nous discuterons en particulier de l'évolution de ces résultats en fonction des critères de découpage choi-

sis. Un des objectifs étant qu'elle suive au mieux la topologie de la scène étudiée. Plusieurs statistiques concernant les graphes de visibilité obtenus terminent cette section et nous montrons concrètement comment ces nouvelles informations peuvent être exploitées dans la visualisation interactive de scènes complexes.

Enfin, la section 5 conclut ces travaux en rappelant les contributions apportées à la résolution des problèmes posés initialement. Etant donné les résultats encourageants obtenus, plusieurs voies sont finalement proposées pour poursuivre ce travail.

2 Etat de l'art

Dans cette section, nous présentons différentes méthodes déjà proposées dans la littérature pour diminuer la complexité du calcul de visibilité dans une scène. Une première partie décrit clairement le problème et les traitements nécessaires pour le résoudre. Les travaux déjà effectués dans cette voie sont présentés dans la partie suivante. La dernière partie de ce chapitre contient une analyse des différentes méthodes.

2.1 Position du problème

Les calculs de visibilité sont primordiaux dans la création d'une image. En effet, pour visualiser correctement une scène, il est nécessaire de déterminer avec précision quelles sont les objets visibles (totalement ou en partie) depuis un point d'observation donné. Dans ce cas, la complexité de ces calculs est liée au nombre d'objets présents dans celle-ci. Ainsi, par exemple, les temps de calcul et la place mémoire nécessaires à la visualisation d'un immeuble tout entier seront naturellement bien supérieurs à ceux demandés par un petit appartement. Mais cette complexité ne dépend pas seulement de la taille d'une scène: si nous choisissons, par exemple, de traverser notre immeuble, l'ensemble des objets visibles peut fortement varier avec notre emplacement (complexité spatiale) ou bien encore selon la dynamique propre à la scène due aux mouvements de certains objets dans celle-ci (complexité temporelle).

Les modèles architecturaux (nos principaux modèles applicatifs) rassemblent facilement ces complexités. Ce sont bien souvent des milieux fortement occlusifs : seule une faible proportion des objets de la scène demeure effectivement visible depuis un point d'observation donné. Ainsi, par exemple, les étages inférieurs (ou supérieurs) à celui occupé par notre observateur lui sont définitivement invisibles, et le nombre d'objets effectivement visibles reste très inférieur au nombre total des objets contenus dans la scène.

Les algorithmes de visibilité qui visent, dans ce cas, à réduire le nombre d'objets à afficher, peuvent exploiter cette propriété. Pour cela, il est possible de structurer la scène en cellules et ouvertures. Une cellule est un volume spatial (vide ou contenant des objets) obtenue par un découpage de la scène. Une ouverture est une région 2D transparente située sur la frontière séparant deux cellules adjacentes. Deux cellules adjacentes sont reliées par une ouverture (figure 1).

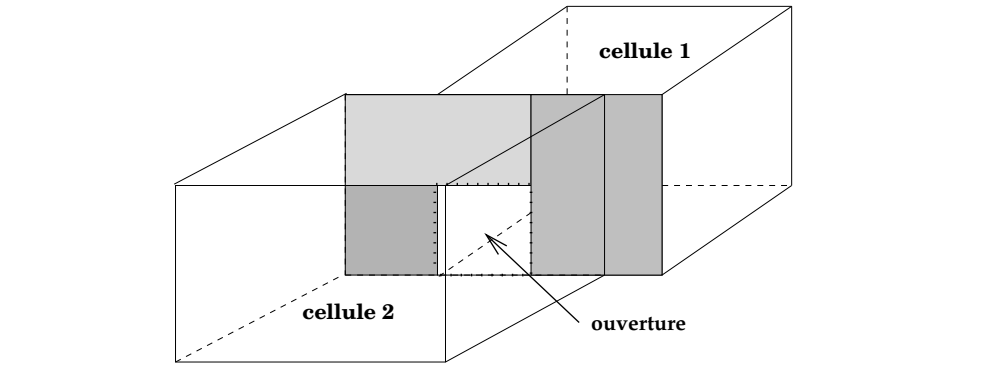

figure 1 : Ouverture entre deux cellules.

Beaucoup de scènes complexes sont naturellement structurées. C'est le cas par exemple des environnements architecturaux (un batiment est constitué d'étages et les étages de pièces). Clairement, le découpage d'un tel modèle s'effectuera de préférence le long des murs, de manière à ce que les cellules correspondent aux pièces de l'immeuble et les ouvertures aux portes et fenêtres. En déterminant les relations de visibilité des cellules entre elles (une cellule n'est visible d'une autre qu'à travers une séquence d'ouvertures), nous accélérons alors fortement la recherche des objets visibles d'un observateur: ils sont nécessairement dans les cellules visibles de la cellule occupée par l'observateur.

De même, les simulations d'éclairage basées sur les échanges d'énergie lumineuse entre les surfaces contenues dans une scène (telle que la radiosité) sont directement concernées par cette structuration de la scène en cellules. Dans la plupart des algorithmes de radiosité, les échanges de rayonnements lumineux sont décrits par l'équation suivante:

$$B(P) = E(P) + \rho_i \sum_{j=1}^{N} \int_{S_j} k(P,P')\, vis(P,P')\, B(P')\, dP' \qquad (1)$$

où :

– $B(P)$ est l'énergie lumineuse émise par un point P;

- $E(P)$ est l'émission propre de P (cas d'une source lumineuse);
- ρ_i est la réflectivité de la facette i; cette valeur est caractéristique du matériau composant la facette i et représente le rapport de l'énergie réémise sur l'énergie reçue;
- N est le nombre de facettes contenues dans la scène;
- dP' est un élément de surface autour d'un point P';
- $k(P, P')$ désigne la proportion d'énergie émise par P' et reçue par P;
- $vis(P, P')$ vaut 1 si les points P et P' sont visibles et 0 sinon.

Pour une longueur d'onde donnée, l'équation (1) indique que la quantité d'énergie lumineuse émise par un point P est égale à la quantité d'énergie émise par P, par auto-émission, à laquelle s'ajoute une proportion de l'énergie lumineuse incidente réfléchie en P.

Dans la plupart des cas, il est impossible de résoudre de façon analytique l'équation (1). Des méthodes numériques doivent être utilisées. Pour cela, il est nécessaire de travailler sur une version discrète de l'équation (1). Celle-ci est obtenue en subdivisant la scène en petites surfaces : les carreaux. Pour faciliter l'expression de l'équation discrète de la radiosité, on suppose que la subdivision est telle que la radiosité est constante en chaque point d'un même carreau.

La radiosité en un point B_i d'un carreau donné s'exprime par l'équation suivante :

$$B_i = E_i + \rho_i \sum_k B_k F_{ik} \tag{2}$$

où :

- l'entier k indice cette fois un ensemble de N carreaux
- ρ_i est la réflectivité du carreau
- F_{ik} est le facteur de forme entre les carreaux i et k c'est-à-dire la proportion d'énergie émise par le carreau i et arrivant sur le carreau k

Pour une bonne approximation de la fonction de radiosité sur ces carreaux, cette subdivision doit être suffisamment fine. Le nombre élevé de carreaux qui en résulte

est la cause de la complexité des calculs de radiosité. En effet, l'opération de base de l'algorithme consiste à transférer de l'énergie entre deux carreaux. Il s'agit donc de déterminer si le carreau récepteur est visible ou non du carreau émetteur, et cette étape nécessite d'effectuer des calculs de visibilité sur la scène (figure 2).

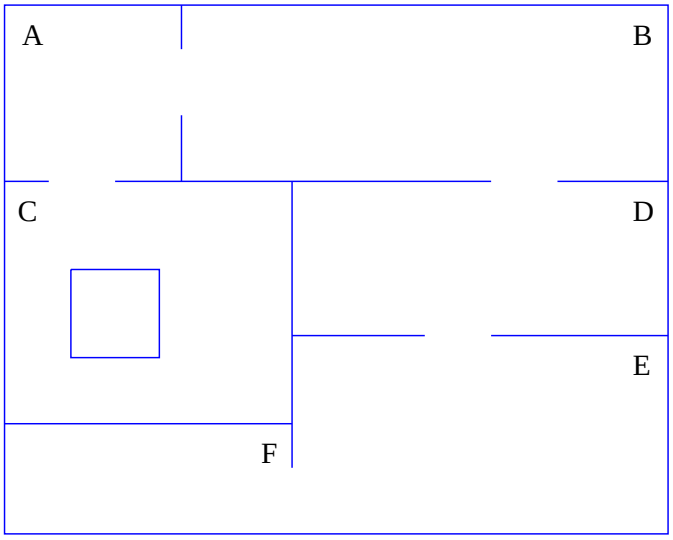

figure 2 : Plan d'un appartement.

Etant donné un point quelconque dans la pièce F, il est évident que seule une faible partie de la scène contribue à son éclairement. Il est possible de tirer parti de cette propriété pour réduire la complexité des algorithmes : on ne tient compte dans la résolution des radiosités que des objets susceptibles d'échanger de la lumière avec un point de F.

Le problème revient donc à déterminer l'ensemble des points de la scène visibles depuis la pièce F. Naturellement, les principaux éléments structurants de la scène (tels que les murs, sols, ou plafonds) sont largement responsables des occlusions pouvant limiter cet ensemble. Le problème de la visibilité de la pièce F se ramène à celui de la visibilité d'un observateur mobile dans cette pièce.

Pour traiter ces problèmes, deux étapes apparaissent finalement nécessaires :

1. structurer la scène en cellules (en tenant compte des objets les plus occlusifs);
2. calculer les relations de visibilité entre ces cellules.

Remarquons la similitude entre la résolution de l'équation de radiosité et d'un déplacement interactif (ou walkthrough) dans une scène. Dans les deux cas, il s'agit d'effectuer des calculs de visibilité entre les points d'une région de l'espace (appartenant à une courbe ou à une surface) et le reste de la scène.

La section suivante présentent les travaux déjà effectués dans cette direction.

2.2 Travaux précédents

Différentes approches pour optimiser les calculs de visibilité existent dans la littérature. Par exemple, dans les algorithmes basés sur le lancer de rayon, plusieurs travaux proposent de structurer la scène. Ainsi, dans [1], Bouatouch et al. appliquent une grille spatiale 3D sur la scène. L'espace est régulièrement subdivisé en boites, ou cellules. Les algorithmes incrémentaux qui en résultent accélèrent de façon importante le lancer de rayon.

Glassner ([2]) propose une hiérarchie de volumes englobants à base d'octree: l'espace est cette fois subdivisé selon une complexité locale. Cette structure permet d'utiliser des algorithmes récursifs pour le lancer de rayon, et procure une représentation mémoire beaucoup plus compacte de la scène.

Dans [3], Thirion utilise une structure de données particulière, nommée Trie, pour combiner les avantages d'une représentation mémoire compacte (meilleure que celle obtenue avec les octree), avec une rapidité accrue des algorithmes de lancer de rayon.

Mais, ces structurations de la scène sont uniquement faites en fonction de la distribution des objets dans l'espace et non des relations de visibilité. Dans certains cas, on doit tenir compte d'un grand nombre d'objets qui ne seront pas visibles. Nous présentons ci-dessous plusieurs travaux exploitant les relations de visibilité entre objets, pour optimiser les calculs d'élimination d'objets cachés.

2.2.1 Jones : une première approche

Dès 1970, Jones [4] est le premier à décrire un algorithme de visibilité basé sur une structuration de la scène en cellules. Malgré les possibilité matérielles très limitées de l'époque (seul l'affichage perpective en fil de fer est réalisé sur des écrans de

faible résolution), le problème de l'élimination des arêtes cachées était déjà largement étudié. Mais, à la différence des travaux de Jones, les solutions proposées ne permettent pas de visualiser aisément, et de manière (quasi-) interactive, un environnement 3D. En effet, les temps de calculs nécessaires pour éliminer ces "arêtes cachées" sont encore trop importants, et les capacités mémoires bien insuffisantes, pour pouvoir envisager la libre progression d'un observateur virtuel dans une scène quelconque.

Pour résoudre ce problème, Jones propose de subdiviser la scène en cellules spatiales délimitées par les facettes des objets, d'une part, et par des faces virtuelles transparentes (qu'il appelle ouvertures) d'autre part. Les cellules ainsi établies doivent être convexes. Les ouvertures ainsi introduites sont également convexes (ce sont des polygones) et partagées par deux cellules adjacentes. Une telle décomposition est réalisée par l'utilisateur et peut-être facilement (selon l'auteur) appliquée à un environnement architectural (figure 3). Les cellules correspondent ainsi quasiment aux pièces et aux ouvertures (représentées en pointillées sur la figure) correspondent pour la plupart aux portes de l'appartement.

La structure représentant l'interconnexion des cellules entre elles est alors un graphe dans lequel chaque noeud représente une cellule et chaque arc une ouverture (figure 3). Partant d'une cellule donnée (un noeud du graphe) et d'une direction de visée, une procédure de recherche récursive est alors engagée sur ce graphe. Il s'agit de déterminer à l'affichage quelles cellules sont visibles de l'observateur. Si nous plaçons, par exemple, un observateur dans la cellule A, celui-ci peut voir les murs de sa cellule ainsi qu'une partie de ceux délimitant les cellules B et C au travers des ouvertures. Plusieurs cellules (noeuds dans le graphe) peuvent ainsi être atteintes via une liste d'ouvertures (liste d'arcs).

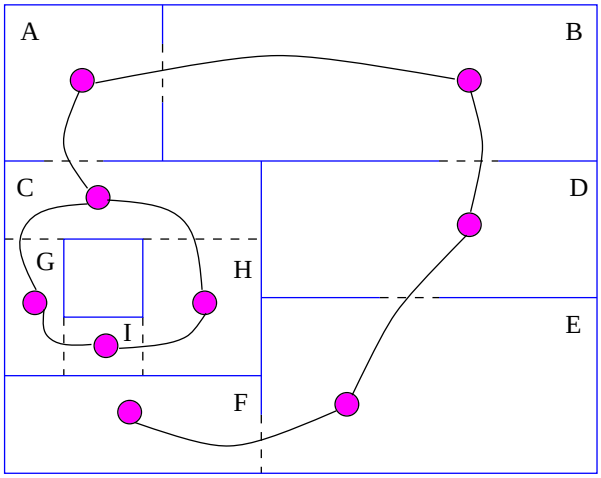

figure 3 : Subdivision spatiale 2D et graphe d'adjacence correspondant.

A l'écran, la projection d'une ouverture forme un contour polygonal servant de masque de visibilité (figure 4). A chaque nouvelle cellule (noeuds du graphe) rencontrée, un nouveau masque de visibilité est calculé. Celui-ci est formé à partir de l'intersection du masque courant et du contour de l'ouverture traversée, projeté sur l'écran. Le masque obtenu permet de déterminer quelles sont les éléments de la cellule atteinte encore visibibles. Ainsi, toutes les lignes situées sur les faces et arêtes intérieures à la cellule (et ne faisant pas partie de l'ouverture traversée) sont projetées sur l'écran et découpées par le masque.

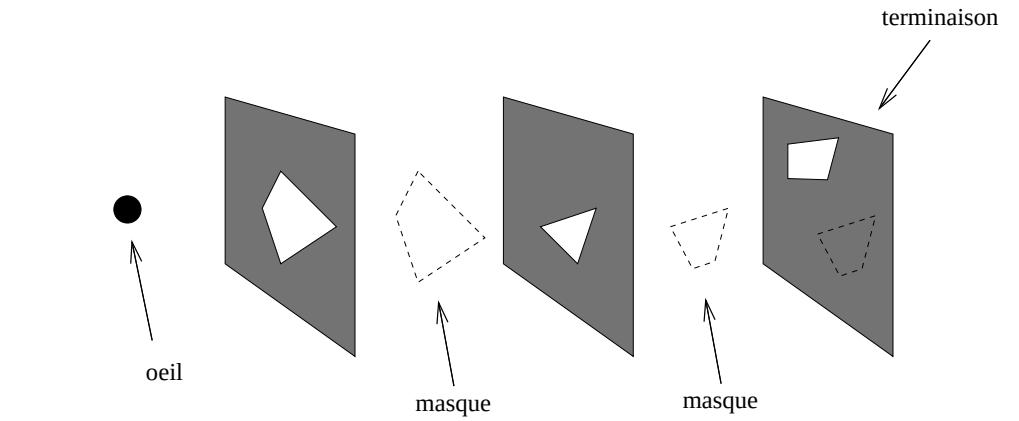

figure 4 : Méthode des masques de Jones.

Si l'intersection est vide, l'ouverture est invisible et la progression dans le graphe s'arrête. L'algorithme "remonte" récursivement à la cellule précédente (en retrouvant également le masque précédent) pour détecter d'autres cellules visibles à travers une nouvelle séquence d'ouvertures.

Si l'intersection n'est pas vide alors une nouvelle cellule est atteinte et le processus de recherche continue avec examinant ses autres ouvertures.

Notons que l'ensemble des noeuds parcourus (cellules visibles) forment un sous-graphe orienté du graphe d'adjacence (figure 5).

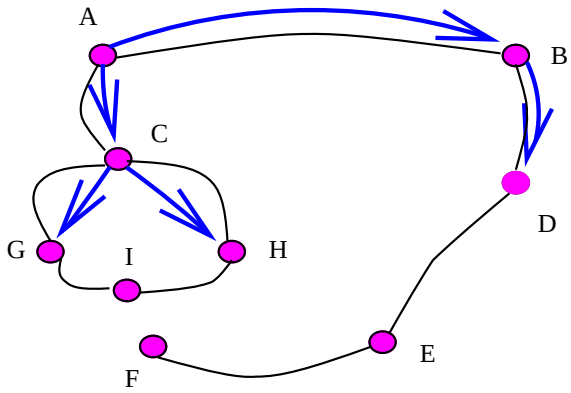

figure 5 : Sous-graphe orienté des cellules visibles.

Cette approche a permis à Jones de générer rapidement, et de manière quasi-interactive, plusieurs séquences de vues d'objets 3D ou d'environnements architecturaux (peu complexes néanmoins).

2.2.2 La structuration automatique de scènes proposée par d'Airey

Avec la progression spectaculaire de la puissance de calcul et des capacités mémoires disponibles sur les machines, des scènes plus complexes peuvent désormais être traitées et visualisées avec réalisme. Le réalisme d'une image dépend en particulier de la richesse des éléments qui la composent (c'est à dire du nombre d'objets dans la scène) et de la précision avec laquelle ils sont modélisés (par exemple, le nombre de facettes qui les composent) et rendus. L'affichage d'une image est directement pris en charge par un pipeline graphique, qui s'occupe de résoudre les problèmes de visibilité.

Néanmoins, un goulot d'étranglement subsiste : il s'agit du nombre maximum de primitives géométriques (des facettes par exemple) que le sous-système graphique permet d'afficher (en respectant les visibilités) par seconde. Or simuler la traversée d'un environnement virtuel nécessite de pouvoir afficher plusieurs images de la scène par seconde. A partir d'une fréquence d'environ 20 images par seconde, les saccades

dans l'affichage disparaissent (Airey évoque un minimum acceptable de 6 images par seconde).

L'objectif des travaux d'Airey [5] est de garder un haut niveau d'interactivité dans la traversée d'un environnement virtuel (ici, un immeuble) sans toutefois pénaliser le réalisme des images affichées. Pour cela, une stratégie de précalcul des relations de visibilité est engagée avant l'étape de la traversée. Ceci permet de diminuer le nombre de primitives que le pipeline graphique doit traiter à chaque mouvement de l'observateur. Airey exploite ainsi plusieurs propriétés caractéristiques des environnements architecturaux:

1. un modèle architectural est moins souvent modifié que les points de vue sur ce dernier, ce qui justifie une étape de précalcul des visibilités avant la traversée;
2. dans la plupart des cas, la majeure partie de la scène ne contribue en rien à une visualisation locale (intérieur d'une pièce dans un immeuble par exemple);
3. beaucoup de polygones sont axiaux (c'est à dire parallèles à deux des trois axes de coordonnées) et rectangulaires;
4. quand un observateur se déplace dans la scène, l'ensemble des polygones visibles varie peu, excepté aux voisinages des ouvertures (portes ou fenêtres par exemple).

La trajectoire possible d'un observateur dans la scène est dessinée (figure 6). La partie de la scène qui lui est visible depuis le point P1 ne changera certainement pas beaucoup (ce sera principalement les murs et les objets éventuels de la pièce F) jusqu'à ce qu'il se trouve au voisinage du point P2. Son passage par l'ouverture (ici une porte) s'accompagne en effet d'un changement radical de l'ensemble des objets qui lui sont visibles: il voit maintenant surtout les éléments composant la pièce E.

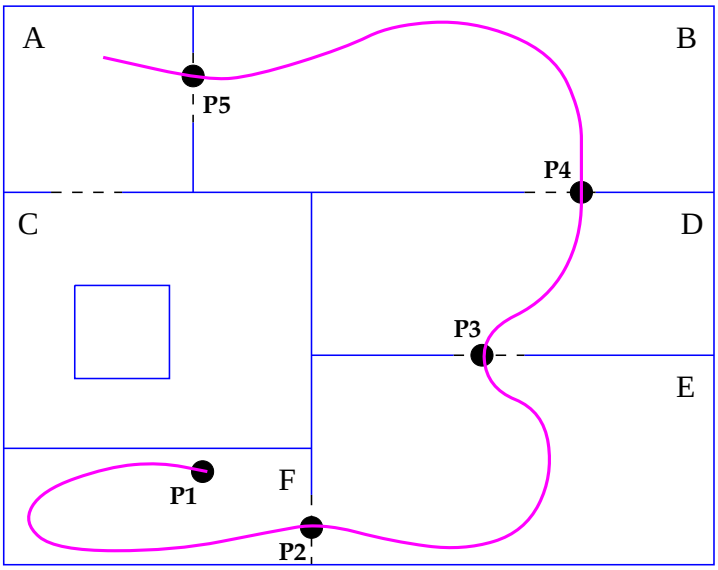

figure 6 : Exemple d'itinéraire emprunté par un observateur.

Airey propose donc de subdiviser le modèle en cellules, pour lesquelles l'ensemble des polygones potentiellement visibles à partir de ces cellules (ou encore susceptibles d'être visibles depuis certains points des cellules) est calculé. Comme la taille de ces ensembles est souvent largement inférieure à la taille du modèle tout entier, le processus d'affichage s'en trouve nettement accéléré. Ce découpage de la scène doit cependant suivre deux objectifs :

1. minimiser la taille des ensembles de polygones potentiellement visibles;
2. minimiser le nombre de cellules obtenues (pour éviter en particulier que deux cellules adjacentes ne se retrouvent avec des ensembles de visibilité quasiment identiques).

De plus, l'algorithme de subdivision doit permettre de retrouver aisément la cellule contenant le point de vue courant. Cette contrainte implique l'utilisation d'une structure de données adaptée à une recherche rapide : Airey choisit donc d'utiliser

des arbres de subdivision binaire[1] (découpage BSP) et perpendiculaires aux axes du repère (pour exploiter la propriété 3). Finalement, l'algorithme choisit les plans de découpage en fonction des critères suivants:

1. Le plan découpe-t-il le modèle de façon équilibrée;
2. Le plan empêche-t-il toute relation de visibilité entre les deux régions obtenues (i.e. quel est son degré d'opacité);
3. combien de polygones sont découpés par le plan (un tel polygone sera en effet visible des deux cotés du plan).

Une combinaison de ces critères est effectuée pour obtenir un ordre de priorité parmi les plans de découpage candidats. Le découpage se termine lorque tous les plans candidats ont été choisis. Le processus génère un arbre dont les noeuds représentent les plans de séparation et les feuilles les cellules. Les plans de découpages (figure 7)sont numérotés selon l'ordre de leur sélection.

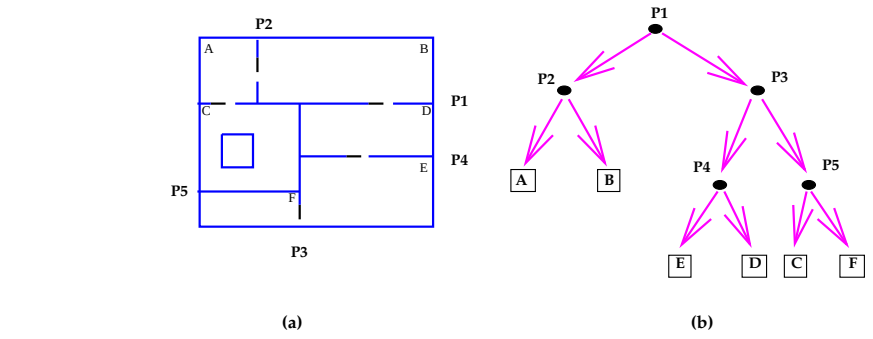

figure 7 : Exemple de décomposition d'une scène (a) et arbre BSP obtenu (b).

Une fois la scène structurée en cellules, les sous-ensembles du modèle, visibles d'un observateur mobile dans chacune des cellules, sont calculés et associés à ces cellules. Ces sous-ensembles comprennent les polygones situés sur la frontière et à

1. cette technique sera détaillée plus loin

l'intérieur des cellules obtenues, ainsi que ceux visibles au travers des ouvertures des cellules. Ce dernier groupe de polygones semble pourtant difficile à calculer de manière exacte. Devant la complexité et l'inefficacité des méthodes existantes pour retrouver les polygones visibles depuis une ouverture, Airey opte pour un algorithme calculant de manière approchée cet ensemble: plusieurs rayons de visibilité sont lancés depuis des points aléatoirement choisis sur les ouvertures. Si un polygone est touché par un rayon, il est alors visible (peut-être partiellement) et automatiquement inclus dans l'ensemble des polygones potentiellement visibles.

Les résultats obtenus par Airey montrent des temps d'affichage accélérés d'un facteur variant de 3 à 30. De plus, l'augmentation de la taille mémoire (centrale) nécessaire pour stocker puis gérer ces précalculs de visibilité pendant l'affichage n'excède pas 20%.

2.2.3 Plantiga et les évènements visuels

Le problème de la traversée interactive de modèles 3D complexes (un immeuble par exemple) intéresse également Plantiga [6]. Partant des mêmes constatations qu'Airey quant aux difficultés du matériel graphique pour visualiser de telles scènes, il opte également pour une phase de précalcul réduisant sensiblement le nombre de polygones à afficher à chaque image en identifiant ceux qui ne seront pas visibles depuis certaines régions de la scène.

Pour simplifier le problème, Plantiga propose, dans un premier temps, de limiter les mouvements de l'observateur à un espace 2D: le plan parallèle au sol est situé au niveau des yeux d'un promeneur potentiel. L'observateur peut, bien entendu, déplacer son regard autour (et même au-dessous ou au-dessus) de lui, mais il ne peut pas s'élever ou descendre de ce niveau d'observation. La phase de précalcul des relations de visibilité préconisée par l'auteur consiste alors à calculer une partition de ce plan en régions où l'apparence topologique reste constante, c'est à dire depuis lesquelles un même ensemble d'objets est visible. Par objets, Plantiga désigne un ensemble de polygones spatialement compacts et non séparés par d'autres faces importantes. Il suppose que la distinction entre les objets et les principaux éléments structurants de la scène (tels que les murs) est établie.

Les objets sont souvent constitués de centaines (voire de milliers) de polygones, et les calculs de visibilité impliquant toutes les faces d'un objet deviennent très vite excessifs. Pour des raisons d'efficacité, une boite englobante, dont les arêtes sont

parallèles aux axes de coordonnées, est calculée pour chaque objet. Les calculs de visibilité sont alors effectués sur cet englobant, plutôt que sur les faces de l'objet. Ainsi, si une boite est invisible depuis un point de vue donné, toutes les faces composant cet objet seront certainement invisibles également. En revanche, la visibilité de la boite implique une visibilité potentielle (et peut-être partielle) de l'objet. Enfin, cette méthode d'approximation de l'objet par une boite englobante autorise la modélisation d'objets par des surfaces courbes sans augmenter la complexité des calculs de visibilité.

L'algorithme identifiant les cellules contenant un même ensemble d'objets potentiellement visibles calcule les évènements visuels: ce sont les points d'observation depuis lesquels l'image change topologiquement s'ils sont traversés. Les évènements visuels apparaissent aux endroits depuis lesquels une arête et un sommet (évènements EV), ou encore trois arêtes (évènements EEE), de la scène semblent s'intersecter. La figure 8 présente deux exemples de tels évènements visuels. Dans le premier cas (évènement EV), la facette triangulaire semble disparaître (ou apparaître) avec le mouvement de l'observateur. Le second cas (évènement EEE) montre que la facette occultée par l'évènement ne contient pas forcément une des arêtes générant l'évènement. Les points d'évènement potentiel forment des lignes de vue coplanaires dans le cas d'évènements EV, et des lignes de vue appartenant à une surface quadrique trouée dans le cas d'évènements EEE. L'intersection de ces surfaces avec le plan d'évolution de l'observateur est un ensemble de segments de droites (pour les évènements EV) et de segments de courbes (pour les évènements EEE) qui partitionnent ce plan en zones 2D dans lesquelles l'ensemble des objets potentiellement visibles ne change pas.

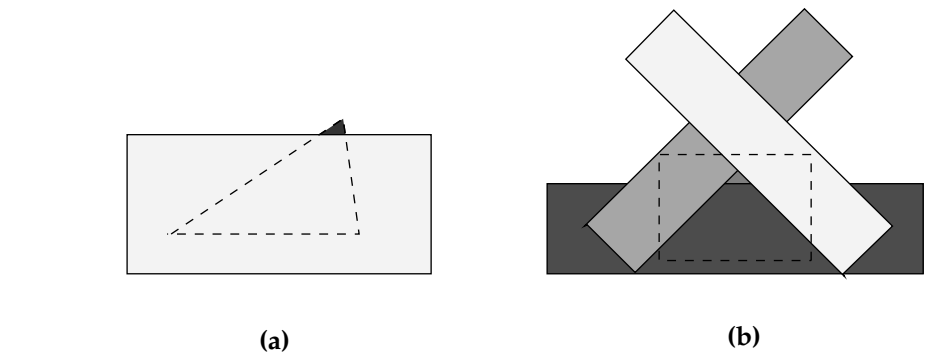

figure 8 : Evènements visuels EV (a) et EEE (b).

Plantiga utilise finalement un graphe d'évènements visuels pour gérer, durant la traversée, ces ensembles de polygones potentiellement visibles: les arêtes du graphe représentent les segments de droites et de courbes de la partition, alors que les sommets représentent les intersections de ces segments. A chaque sommet, les coordonnées des points d'intersection (dans le plan d'évolution) sont stockées. A chaque arête les constantes représentant la courbe quadratique du segment concerné sont calculées, et la liste des objets à ajouter, ainsi que celle des objets à enlever, sont établies.

Ainsi, lorsque l'observateur se déplace, il devient possible de déterminer les objets qui deviennent invisibles ou potentiellement visibles en vérifiant s'il quitte la cellule courante. Dans ce cas, il suffit d'effectuer une simple mise à jour de la liste des objets à afficher en fonction des listes d'ajout et de retrait présentes sur l'arête du graphe représentant le segment de frontière traversé.

Cette approche a permis à Plantiga de visualiser la traversée de grosses scènes pouvant notamment contenir des surfaces courbes. La partition de l'espace des points de vue (le plan d'observation) en cellules est automatiquement générée à partir d'une sélection des objets et des murs faites sur la scène. Les ensembles de polygones potentiellement visibles précalculés sont conservatifs (ils contiennent au minimum l'ensemble des polygones effectivement visibles) et c'est le pipeline graphique qui calcule en dernier ressort les visibilités exacts.

2.2.4 Les travaux de Teller

Les différentes publications de Teller [7, 8, 9, 10] proposent également de structurer les scènes architecturales pour traiter des problèmes de visibilité. Cependant, les algorithmes présentés s'adressent non seulement au problème de leur traversée interactive, mais également aux calculs d'illumination globale de telles scènes (comme la méthode de radiosité par exemple).

Constatant la difficulté à déterminer exactement l'ensemble des objets visibles depuis un point d'observation donné, Teller choisit plutôt de précalculer un surensemble de ces objets. Cette première phase entraine la structuration de la scène en cellules spatiales convexes séparées par des murs. Tout comme Plantiga, il suppose que la représentation du modèle distingue clairement les murs (ou principaux éléments occlusifs de la scène) des autres objets.

La décomposition utilise un arbre BSP (Binary Space Partitioning) et se base (comme le découpage d'Airey) sur le degré d'opacité des polygones dans la scène (cf. définition plus loin). L'algorithme part d'une cellule initiale correspondant à la boite englobante de tous les polygones de la scène, exceptés ceux décrivant les objets non occlusifs. Un plan de découpage candidat est alors celui dont l'aire totale des polygones supportés dépasse une certaine fraction de l'aire du polygone résultant de l'intersection du plan avec la cellule courante (figure 9). La section s1 (respectivement s2) est le polygone résultant de l'intersection entre la cellule et le plan supportant le polygone P1 (respectivement P2). Le plan supportant le polygone P1 (resp. P2) est alors candidat comme plan de découpage si l'aire de P1 dépasse une certaine fraction (par exemple 30 %) de l'aire de s1 (resp. s2).

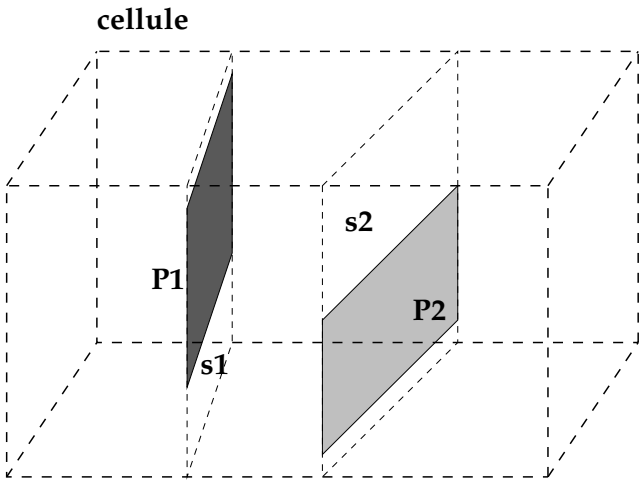

figure 9 : Plans de découpage d'une cellule.

Une fois la décomposition en cellules achevée, les ouvertures situées sur les frontières des cellules sont calculées. Finalement, la subdivision spatiale est représentée par un graphe d'adjacence sur ces cellules tel celui présenté dans la figure 10. Deux cellules sont ainsi reliées s'il existe une ouverture entre elles.

A partir du graphe d'adjacence des cellules, Teller calcule alors pour chaque cellule l'ensemble des cellules qui lui sont visibles. Cette opération nécessite de parcourir récursivement le graphe d'adjacence (partant d'un noeud donné) en déterminant à chaque nouveau noeud traversé, si la cellule correspondante est visible (souvent partiellement) ou non de la cellule de départ. Une cellule ne peut "voir" ses voisines qu'à travers ses ouvertures, et les cellules plus éloignées qu'au travers d'une séquence d'ouvertures. Ainsi, s'il existe un rayon traversant cette séquence d'ouvertures sans rencontrer d'obstacle alors les cellules situées aux deux extrémités sont visibles. La figure 10 montre l'existence d'un tel rayon traversant une séquence d'ouvertures.

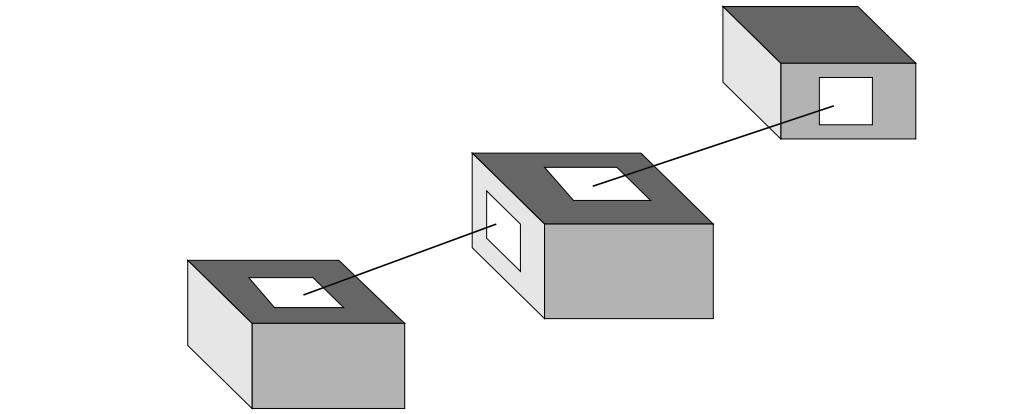

figure 10 : Rayon de visibilité traversant une liste d'ouvertures.

Dès lors que la visibilité entre deux cellules est établie, Teller construit les volumes effectivement visibles dans chacune des cellules. En effet, si deux cellules sont visibles à travers une séquence d'ouvertures, seule une portion de l'une d'elle est visible de l'autre (et réciproquement). Cette dernière remarque est illustrée par la figure (figure 11). Les cellules B, D et E sont les cellules atteintes pendant la recherche des cellules visibles depuis la cellule F. Mais seules les zones grisées de ces cellules seront visibles d'un observateur circulant dans la cellule F. Connaissant ces volumes de visibilité, il est alors facile d'identifier les objets potentiellement visibles depuis une cellule : ceux qui intersectent ces mêmes volumes.

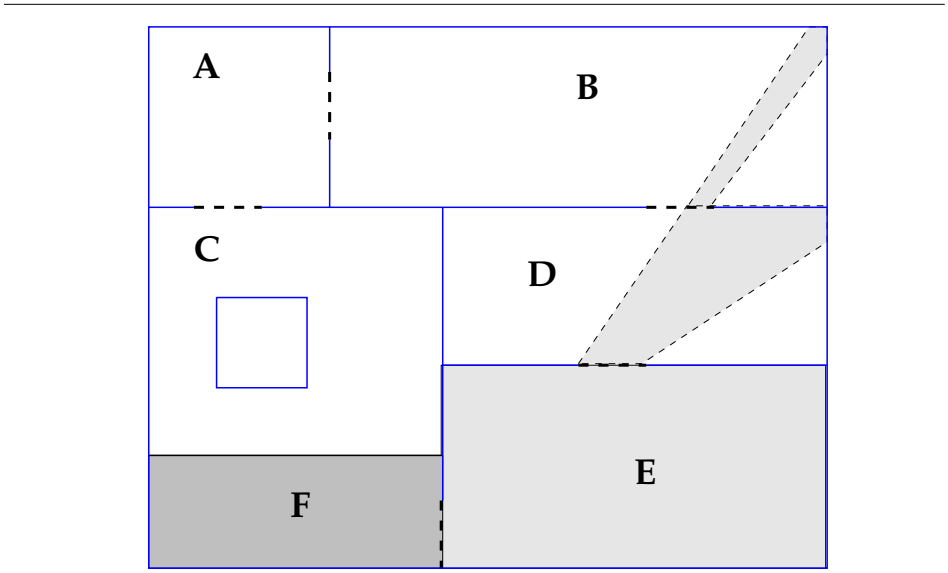

figure 11 : Regions visibles depuis une cellule.

Tous ces calculs correspondent à une étape de prétraitement des calculs de visibilité d'une scène donnée. Durant la phase dynamique (déplacement d'un observateur dans la scène), l'algorithme de visualisation détermine la cellule dans laquelle se trouve l'observateur. Connaissant cette cellule, les cellules (et même les parties de ces cellules) potentiellement visibles de celle-ci sont automatiquement retrouvées. L'ensemble des polygones que doit traiter le pipeline graphique se trouve donc largement réduit puisqu'il ne contient plus que les polygones qui appartiennent aux parties des cellules visibles (murs et objets), et non pas tous les polygones de la scène.

Pour accélérer les calculs d'illumination globale (et notamment le calcul des radiosités) dans de tels environnements, Teller propose de pousser un peu plus loin la phase de précalcul des visibilités. Rappelons que la méthode de radiosité repose sur un découpage des surfaces de la scène en carreaux. Pour calculer correctement les échanges d'énergie lumineuse entre ces carreaux, il est nécessaire de savoir si des oc-

clusions existent entre eux, c'est à dire d'effectuer des calculs de visibilité sur la scène.

Pour réduire le coût de ces calculs, Teller identifie tout d'abord les paires de polygones mutuellement visibles dans la scène. Cette opération s'effectue en partant des informations de visibilité inter-cellules déjà calculées. Pour chaque polygone d'une cellule, l'auteur calcule le volume de la scène qui lui est visible en traversant les séquences d'ouvertures établies lors du calcul des visibilités inter-cellules. Quand une cellule est atteinte (en progressant dans le graphe), seuls les polygones de la cellule situés dans ce volume peuvent être visibles. Si la cellule atteinte est vide (i.e. dépourvue d'objets), ces polygones sont entièrement visibles du polygone de départ. Dans le cas contraire, certains polygones peuvent être partiellement ou complètement cachés par les objets. Finalement, cette phase de précalcul des visibilités produit pour chaque polygone P de la scène un ensemble de polygones visibles V(P). Puis, pour chaque polygone R appartenant à V(P), une liste de bloqueurs B(P,R), contenant tous les polygones cachant potentiellement la visibilité des polygones P et R, est construite. C'est cette dernière liste qui est exploitée puis mise à jour dynamiquement pendant la phase de calculs des échanges lumineux entre les polygones de la scène.

En effet, dans l'algorithme de radiosité hiérarchique [11] (utilisé par Teller), quand l'échange lumineux entre deux carreaux ne peut être établi sous un certain seuil d'erreur global, un des deux carreaux de l'interaction est automatiquement subdivisé en carreaux plus petits. Cette erreur est causée par l'existence d'occlusions (partielles) dans la visibilité des carreaux entre eux. Les occlusions sont elles-même la conséquence de la présence de bloqueurs (d'autres carreaux de la scène) entre les carreaux. Les carreaux fils issus du carreau subdivisé peuvent également échanger de l'énergie avec l'autre carreau. Il convient donc à nouveau de connaître les bloqueurs éventuels de ces nouvelles interactions. Ces nouveaux ensembles de bloqueurs sont nécessairement des sous-ensembles des bloqueurs existant entre les carreaux initiaux.

Trois étapes de reclassification des interactions entre carreaux après subdivision sont représentées (figure 12). La partie (i) montre l'interaction initiale considérée à savoir deux carreaux dont la visibilité mutuelle est partiellement cachée par un bloqueur. Comme l'erreur causée par ce bloqueur dans leurs échanges lumineux est trop importante, l'un des deux (ici, celui de gauche) est subdivisé en carreaux plus petits (partie (ii)). La visibilité résultante entre l'un des carreaux fils et l'autre carreau est encore partielle, et l'erreur commise est encore trop grande. Une deuxième subdivision (partie (iii)) augmente finalement le nombre de liens de visibilité totale.

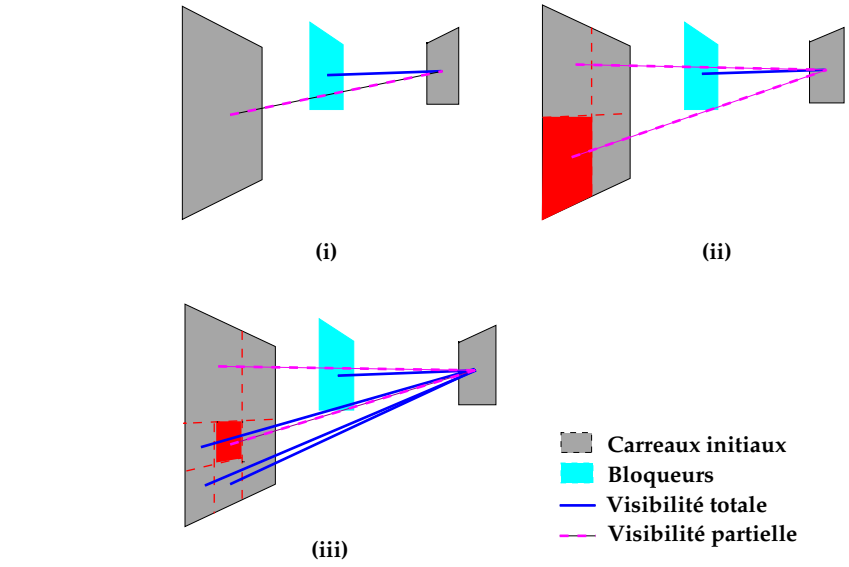

figure 12 : Reclassification des interactions après subdivision.

Les algorithmes présentés par Teller pour établir toutes ces informations de visibilité (visibilité inter-cellules, inter-polygones, listes de bloqueurs) ont permis d'optimiser sensiblement les subdivisions nécessaires pour établir correctement les transferts d'énergie lumineuse entre les carreaux. Ses travaux permettent ainsi un calcul plus rapide des radiosités dans un environnement polygonal complexe.

2.2.5 La mise en oeuvre de Luebke

L'algorithme de Teller ne parait pas suffisament efficace à Luebke dans le cadre d'une visualisation interactive d'un environnement complexe. Dans [12], Luebke et al. décrivent une méthode simple pour évaluer rapidement les polygones potentiellement visibles lors de la visualisation dynamique d'un modèle architectural. Leur approche est similaire à celle de Jones dans le sens où, partant d'une structuration de la scène en cellules et ouvertures, ils déterminent seulement à l'étape de visualisation les cellules visibles depuis le point d'observation.

La décomposition de la scène en cellules et ouvertures est directement intégrée à la phase de modélisation de la scène. Une cellule est donc représentée par un regroupement de polygones, et n'est plus forcément convexe. Les ouvertures correspondent à des polygones particuliers qui contiennent (en plus des informations géométriques les décrivant) le nom des cellules ainsi connectées. Enfin, les objets modélisés appartiennent automatiquement à une cellule de la scène.

Pour évaluer dynamiquement l'ensemble des polygones potentiellement visibles d'un point, les auteurs travaillent directement sur leur projection (et selon la direction de visée de l'observateur) dans l'espace écran. Partant d'une cellule donnée (la cellule contenant l'observateur), chacune de ses ouvertures est ainsi projetée sur le plan image. La projection d'une ouverture sur l'écran donne un polygone dont une boite englobante 2D peut être calculée. Le rectangle obtenu, appelé rectangle de visibilité, sert à identifier les objets des cellules voisines qui ne seront pas visibles: ce sont ceux dont la projection (de leur englobant) sur le plan image tombe en dehors du rectangle. A chaque ouverture traversée, un nouveau rectangle de visibilité est calculé en prenant l'intersection du rectangle de visibilité courant avec celui résultant de la projection de l'ouverture sur l'écran. La visibilité des objets de la nouvelle cellule atteinte est donc testée avec ce nouveau rectangle de visibilité. Comme un même objet peut être visible à travers différentes séquences d'ouvertures, les objets sont marqués dès qu'ils sont affichés à l'écran, évitant ainsi de les réafficher plusieurs fois.

Cet algorithme de visibilité a permis d'écarter 20 à 50 % des polygones d'une scène pour visualiser sa traversée. Même si ces résultats dépendent étroitement du modèle et de la trajectoire empruntée pendant sa visualisation, l'algorithme présenté reste néanmoins une technique d'accélération simple et efficace.

3 Notre approche

La simulation d'éclairage par la méthode de radiosité consiste à calculer les échanges énergétiques entre toutes les surfaces d'une scène. Cette technique prend en compte les contributions de réflexion diffuse de tous les éléments mis en jeu. Elle permet donc d'évaluer avec précision l'éclairage global de la scène.

En revanche, les temps de calcul nécessaires à l'évaluation de la radiosité restent importants. Malheureusement, cette observation est applicable a tous les types de scènes. Il devient donc impératif de prendre en compte le nombre de facettes sur lesquelles nous choisissons d'effectuer ces calculs. En effet, plus les scènes comportent de polygones, plus les calculs sont longs. De plus la sollicitation de la mémoire, très importante, est décuplée par les maillages effectués.

Dans le cas de scènes architecturales complexes, de grandes zones sont cachées par des objets très occlusifs comme des murs par exemple. Par conséquent, un observateur situé dans une pièce d'un immeuble ne voit qu'une faible partie du bâtiment. Afin d'éviter la multiplication des calculs de visibilité, l'option adoptée dans de nombreux travaux est la structuration de la scène.

La décomposition la plus fréquemment utilisée est une subdivision en arbre BSP (Binary Space Partitioning) [14]. Cette méthode consiste à subdiviser un volume en deux parties, par un *plan de découpage*. Chacun des volumes ainsi obtenu est appelé *cellule*. La subdivision se poursuit récursivement tant qu'il reste des plans de découpage.

Deux cellules sont visibles à travers une ou plusieurs *ouvertures*. Ainsi, les regroupements de facettes en cellules forment des *ensembles de polygones potentiellement visibles* ou PVS [22].

En prenant des plans de découpage représentant les murs de la scène, nous obtenons des cellules ressemblant aux pièces de la scène.

Nous proposons d'effectuer une étude sur les critères à utiliser pour le choix des plans de découpage.

Nous verrons tout d'abord quelles sont les hypothèses que nous avons posées. Puis nous présenterons les principes de notre méthode et les critères qui nous ont paru les plus probants. Nous exposerons ensuite un certain nombre de cas particuliers et la manière dont nous les avons traités. Nous verrons enfin comment interpréter les résultats obtenus.

3.1 Hypothèses de travail

Notre étude porte ici exclusivement sur des environnements architecturaux. Néanmoins, ceci ne signifie pas que les études présentés ne soient pas applicables à d'autres types de modèle. Nous avons choisi de traiter un éventail très large de scènes axiales, c'est à dire des scènes où les surfaces ont des normales parallèles à l'un des trois axes du repère utilisé. Dans le cas architectural, la restriction est moindre. En effet, les bâtiments sont composés de pièces rectangulaires dans la plupart des cas.

Néanmoins, par souci de rigueur, nous avons posé les hypothèses suivantes.

Objets non typés
Le type des facettes traitées (tables, sols, etc.) n'est pas pris en compte. Ainsi, nous nous donnons la possibilité de structurer la scène selon tous les polygones sans privilégier les murs ou les sols.

Hiérarchie d'objets à un seul niveau
Nous souhaitons traiter des facettes sans tenir compte de leur éventuel regroupement en objets.
Cette hypothèse nous permet de traiter plusieurs formats de fichiers différents. Nous n'avons pas souhaité travailler avec un modeleur ou un type de fichier particulier, mais plutôt sur un format général.

Polygones rectangulaires
Afin de simplifier les calculs, nous supposons dans la suite de ce rapport que les facettes traitées sont des polygones plans et rectangles.

Orientation des facettes
Les polygones représentant les murs de la scènes sont supposés axiaux, autrement dit, ils sont orientés selon les 3 axes de la scène (figure 13). Nous appelerons *plans axiaux* les plans contenant ces polygones.

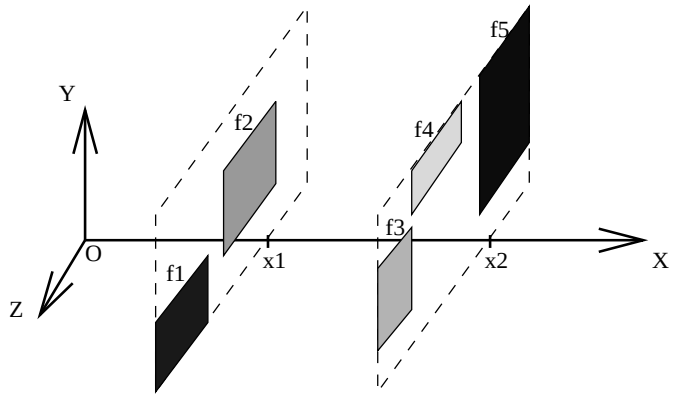

figure 13 : Facettes axiales.

Cette hypothèse ne contraint pas outre mesure les descriptions de scènes architecturales, en effet, la plupart des bâtiments sont à angles droits.
Les autres objets de la scène subissent un traitement différent. Ni les petits objets, ni les facettes non axiales ne sont éliminées. Nous verrons comment les prendre en compte dans le traitement.

3.2 Objectifs à atteindre

Un découpage permettant d'obtenir des cellules équivalentes aux pièces d'une scène est satisfaisant car le nombre d'objets ou de facettes est raisonnable pour effectuer les calculs de radiosité.
Pour parvenir à ces résultats, nous avons choisi de structurer la scène à l'aide d'un arbre BSP (figure 14).

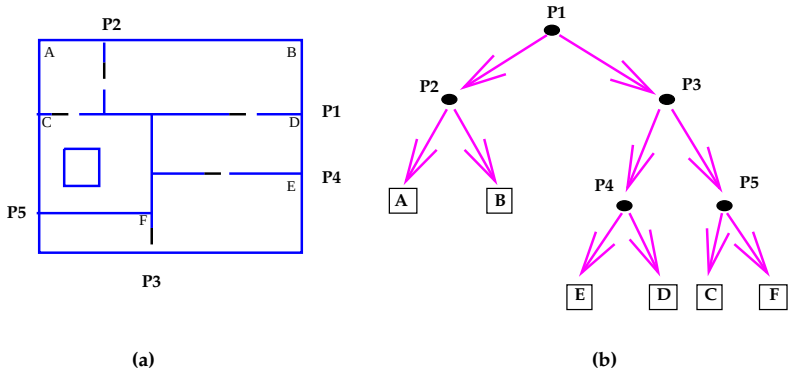

figure 14 : Décomposition BSP d'une scène axiale.

Dans cette figure, les plans P1, P2, P3, P4, et P5 sont les plans de découpage. Nous avons nommé A, B, C, D, E et F les cellules résultantes.

A chaque étape, le plan de découpage doit être judicieusement choisi, afin de respecter au mieux les propriétés suivantes :

Les cellules sont proches des pièces de la scène
Les pièces d'un bâtiment constituent déjà des regroupements d'objets. Une cuisine par exemple regroupe tous les ustensiles nécessaires à la sustentation. Le shéma suivant (figure 15) montre un exemple de bâtiment. Les pièces sont séparées par des murs et reliées par des ouvertures. Nous pouvons intuitivement structurer cette scène en cellules. Soient C1, C2, C3, C4 et C5 les cellules. O1, O2, O3 et O4 sont des ouvertures.

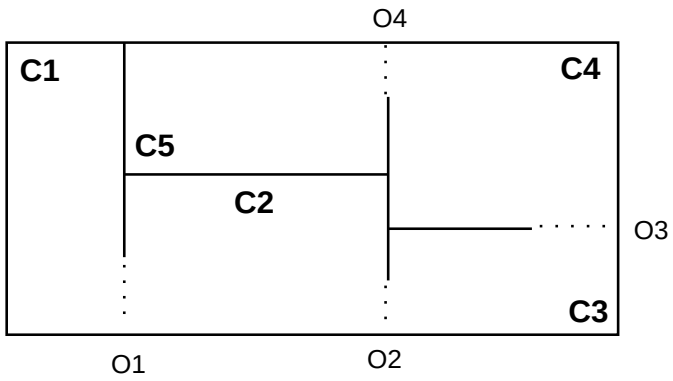

figure 15 : Découpage en cellules.

L'arbre BSP est bien équilibré

Cette propriété est primordiale lorsqu'on souhaite parcourir l'arbre de manière rapide et efficace.

Le nombre de facettes découpées est réduit

Le plan choisi pour la subdivision intersecte souvent un grand nombre de facettes de la scène. Ces polygones, sont soumis à un découpage et insérés dans l'arbre BSP. Ainsi, la convergence est ralentie, le nombre de feuilles est augmenté. Tenir compte de ce critère nous permet de limiter les calculs dus au découpage.

3.3 Principe de la méthode

Une cellule *racine* est créée au départ de l'algorithme. Cette cellule englobe la totalité de la scène.

Les surfaces de la scène sont réparties selon 4 listes.

- Les surfaces dont la normale est parallèle à l'axe Ox,

- Celles dont la normale est parallèle à l'axe Oy,
- Celles dont la normale est parallèle à l'axe Oz,
- Les autres surfaces, non candidates au découpage.

Une *position axiale* est la distance entre l'intersection du plan axial avec l'axe traité et l'origine de ce même axe. Sur la figure 13, la position axiale x1 (resp. x2) des facettes f1 et f2 (resp. f3, f4 et f5) est obtenue en effectuant l'intersection entre le plan axial supportant ces facettes et l'axe perpendiaculaire.

Ainsi, une position axiale nous permet d'obtenir l'ensemble des facettes situées sur un même plan de découpage. Au cours de la subdivision, nous appelons *section* la surface de l'intersection entre ce plan et la cellule courante (figure 16).

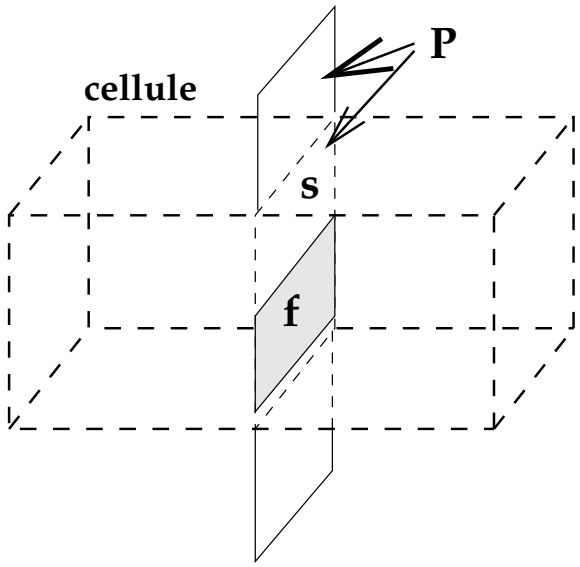

figure 16 : Degré d'opacité $= \frac{aire(S)}{aire(f)}$.

Dans cette figure, f est un polygone axial, supporté par un plan axial P. S est la section de ce plan dans la cellule. Le *degré d'opacité* est le rapport entre l'aire de f et l'aire de la section S.

Nous proposons une méthode de subdivision de type BSP en deux passes principales :

Selection des plans potentiels de découpage

Découpage BSP à partir des plans obtenus et selon un critère donné

L'algorithme utilisé est le suivant.

```
Découpage_BSP(ARBRE arbre, REEL seuil)
{
    PLAN plan_découpage;
    LAXIAL axe[3];

    /* construction des 3 listes axiales en X, Y, et Z */
    axe[0] = Calcule_Liste_Axiale_x(arbre);
    axe[1] = Calcule_Liste_Axiale_y(arbre);
    axe[2] = Calcule_Liste_Axiale_z(arbre);

    /* seuls les plans dont l'aire est supérieure */
    /* à seuil sont candidats au découpage */
    Préfiltrage(axe, seuil);


    /* Un plan est choisi en fonction du critère de découpage */
    /* parmi les plans retenus après préfiltrage */
    plan_découpage = Choix_Plan_découpage(axe);

    /* si un plan de découpage existe alors */
    /* les fils sont créés récursivement */
    si plan_découpage != NULL alors {
        Créer_Fils_droit(arbre);
        Créer_Fils_gauche(arbre);
        Découpage_BSP(arbre.fils_gauche, seuil);
        Découpage_BSP(arbre.fils_droit, seuil);
    }
}
```

La structure PLAN est un tableau de 4 entiers correspondant respectivement aux coefficients a, b, c et d de l'équation $ax + by + cz + d = 0$.

LAXIAL est une liste de facettes dont la normale est parallèle à l'un des trois axes.

ARBRE contient le volume de la cellule, un pointeur sur son fils droit, un pointeur sur son fils gauche et la liste des polygones contenus si c'est une feuille.

seuil est une valeur réelle définie par l'utilisateur. C'est le seuil inférieur du degré d'opacité lors du préfiltrage.

Les fonctions *Calcule_Liste_Axiale_x, Calcule_Liste_Axiale_y, Calcule_Liste_Axiale_z* déterminent les plans de découpage potentiels pour chaque axe du repère. La création de ces plans de dcoupage est réalisée à partir des facettes axiales de la scène. Elles sont triées puis regroupées dans des listes. A chaque distance à l'origine de l'axe correpond une liste de facettes.

La fonction *Préfiltrage* parcourt les trois listes axiales l'une après l'autre pour ne retenir que les plans de découpage les plus occlusifs. La somme des aires des facettes contenues par chacun de ces plans est supérieure à *seuil*. Cette fonction constitue la première phase de l'algorithme.

Choix_Plan_Découpage effectue le choix du plan de découpage. Ce choix dépend du critère de découpage utilisé. Ce critère est appliqué aux trois listes axiales obtenues après *Préfiltrage*. Rappelons que les critères choisis sont :

- l'opacité;
- l'équilibrage de l'arbre;
- le découpage du minimum de facettes.

Dans le cas de la méthode Airey, les trois critères sont utilisés simultanément, chacun étant affecté d'un coefficient de pondération.

3.3.1 Première passe

Dans cette partie de l'algorithme, l'objectif est de conserver les facettes les plus occlusives telles que les murs, certains meubles, les plafonds, etc. Ces facettes sont retenues pour le découpage de la scène en cellules [7]. Les facettes moins occlusives n'interviennent pas dans le partitionnement. Cette méthode permet d'éviter les découpages provoqués par des polygones de petite surface. Un seuil d'opacité est proposé par l'utilisateur. Nous retiendrons comme candidats tous les plans dont le

degré d'opacité (figure 16) est supérieur à ce seuil. Cette première phase permet de réduire considérablement le nombre de polygones de découpage potentiels suivant chaque axe. Ainsi, nous obtenons 3 listes contenant chacune des polygones potentiels de découpage.

3.3.2 Seconde passe

Dans cette partie, le découpage de la scène en arbre BSP est effectué. La difficulté de l'opération réside dans le choix du plan de découpage.
L'objectif principal est bien sûr de diminuer le nombre de polygones potentiellement visibles par cellule. Plusieurs critères permettent d'obtenir de *bons* découpages.
Les critères que nous avons retenus sont les suivants :

Occlusion
Ce critère permet de sélectionner les facettes les plus occlusives pour le découpage de la cellule courante. Les polygones concernés sont dans la plupart des cas des murs, un sol ou des objets de grande taille. Grâce à ce choix, les cellules obtenues sont proches des pièces de la scène.

Plan médian
Seul un arbre BSP bien équilibré permet une optimisation des opérations de recherche et d'insertion de facettes. Lorsqu'un plan de découpage est choisi, deux cellules filles sont créées. Les plans de la cellule mère sont répartis dans les deux filles. Si à chaque étape de la structuration, le nombre de plans potentiels de découpage est identique dans les deux cellules filles, alors l'arbre est bien équilibré. Dans la plupart des cas, le plan respectant cette condition se trouve aussi au centre de la cellule.

Plan découpant le minimum de facettes
Lors de la subdivision d'une cellule en deux filles, un certain nombre de polygones ont une intersection non nulle avec le plan de découpage. Toutes ces facettes sont découpées et chaque partie est insérée dans l'une des cellules filles. Ainsi, le nombre de polygones total de la scène augmente, ainsi que la profondeur de l'arbre.
C'est pourquoi nous proposons un critère de découpage minimisant le nombre de polygones ayant à subir ce traitement.

Nous avons choisi de laisser l'utilisateur fixer le critère de découpage. En d'autres termes, le même critère est appliqué à chaque étape de la décomposition.

Combinaison linéaire de critères.
Airey propose d'effectuer une combinaison linéaire des trois critères cités précédemment [22]. Les coefficients nous permettant d'obtenir les résultats les plus intéressants sont à définir. Airey a choisi ses critères de manière empirique. Les valeurs obtenues sont les suivantes :

0.5 pour le plan d'occlusion maximale;

0.3 pour le plan médian;

0.2 pour le plan découpant le moins de facettes.

D'après les résultats que nous avons obtenus, ses choix sont justifiables. Nous verrons comment les expliquer dans la suite.

3.4 Traitements annexes

Le traitement décrit précédemment ne nous permet pas de prendre en compte les petits objets ni les facettes non axiales. De plus, pour effectuer la simulation d'éclairage, les relations de visibilité entre les cellules sont à définir.

3.4.1 Insertion des facettes non axiales

Lorsque la structuration de la scène est effectuée, les facettes n'ayant pas participé aux découpages sont redistribuées dans leurs cellules respectives. Cette opération nécessite un parcours de l'arbre BSP obtenu.

Certains polygones appartiennent simultanément à deux cellules. Nous avons choisi de les découper en deux parties, selon le plan de découpage de la cellule.

3.4.2 Graphe de visibilité

De plus, ces calculs ne suffisent pas pour la mise en oeuvre la méthode de radiosité. En effet, les relations de visibilité entre cellules sont à préciser.

Notre méthode consiste à établir un graphe d'adjacence (figure 17), puis un graphe de visibilité à partir de l'abre BSP obtenu.

Le graphe d'adjacence représente les relations de voisinage entre les différentes cellules.

- Un sommet du graphe représente une cellule;
- Un arc entre deux noeuds représente une ouverture permettant de passer d'une cellules à l'autre (figure 17).

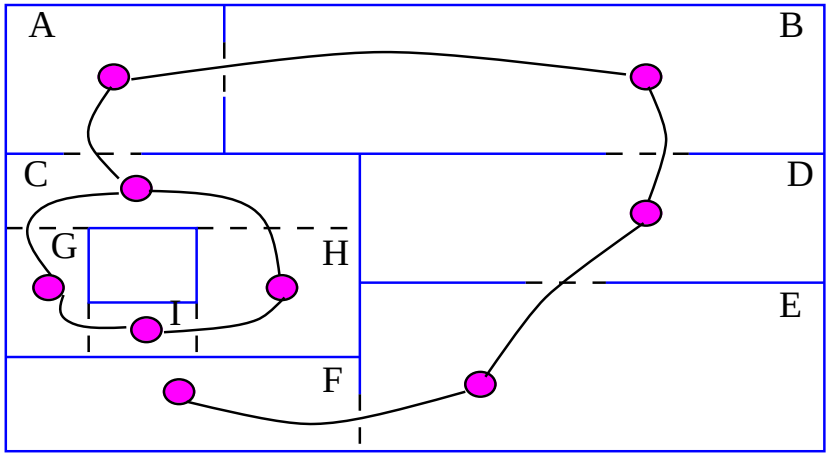

figure 17 : Graphe d'adjacence.

Dans cette figure, les relations d'adjacence sont représentées par un graphe. Ainsi, la cellule A est en relation avec les cellules B et C. La cellule C est en relation avec les cellules A, G et H.

Lorsque le graphe de visibilité est établi, l'ensemble des cellules visibles depuis une cellule donnée peut être calculé en parcourant le graphe d'adjacence.

Pour que deux cellules A et B soient visibles, leurs ouvertures OA et OB doivent êtres visibles l'une de l'autre, totalement ou partiellement. Pour effectuer ce calcul de visibilité, les ouvertures OA et OB sont échantillonnées de manière aléatoire en un certain nombre de points A_i et B_i respectivement. Des rayons joignant A_i et B_j sont lancés (figure 18). Si au moins l'un d'entre eux n'intersecte aucun objet de la

scène, alors les deux cellules sont déclarées visibles l'une de l'autre et un arc est crée dans le graphe de visibilité. Sinon, on estime que les deux cellules ne sont pas visibles.

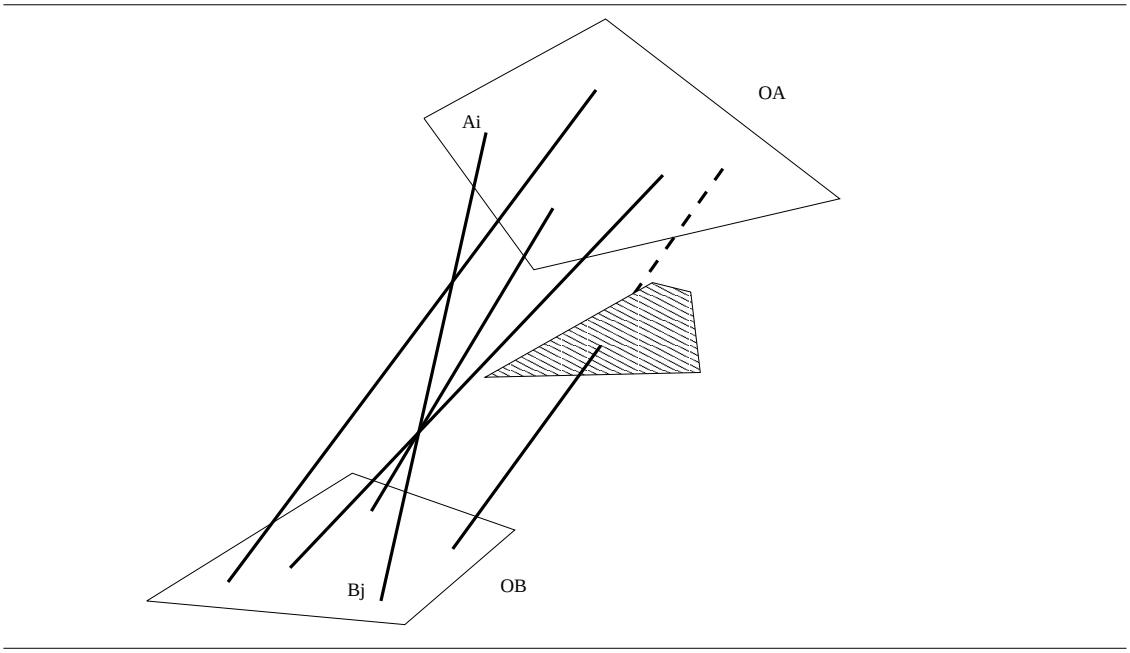

figure 18 : visibilité entre ouvertures

Lorsque les calculs de visibilité sont terminés, nous obtenons un graphe de visibilité où :

- Un noeud représente une cellule;
- Un arc relie deux noeuds lorsque les deux cellules correspondantes sont visibles entre-elles.

Pour chaque cellule, nous obtenons donc un PVS (ensemble de polygones potentiellement visibles) considérablement réduit par rapport à la taille de la scène initiale. Par une séquence d'ouvertures, de nombreuses cellules sont visibles entre

elles. Un graphe de visibilité nous permet d'évaluer ces relations.

Dans la figure précédente, pour chaque cellule, nous définissons une liste de pointeurs sur ses cellules visibles.

3.5 Traitement des cas particuliers

Le découpage BSP nous permet d'obtenir une structuration de la scène en cellules. Ces dernières sont ensuite reliées entre elles par un graphe de visibilité.

La structuration de la scène entraine un certain nombre de cas particuliers.

Néanmoins, ces problèmes ne sont pas insolubles.Nous avons été confrontés à trois cas. Teller propose une solution pour deux d'entre eux [8].

3.5.1 Les cellules trop grandes

Ce problème peut intervenir lorsque les pièces d'un bâtiment sont très grandes et très peuplées. Il n'existe plus de polygone potentiel de découpage malgré la taille de la cellule. La taille des PVS et le nombre d'ouvertures étant proportionnels au volume des cellules, nous perdons l'optimalité de la structuration (figure 19).

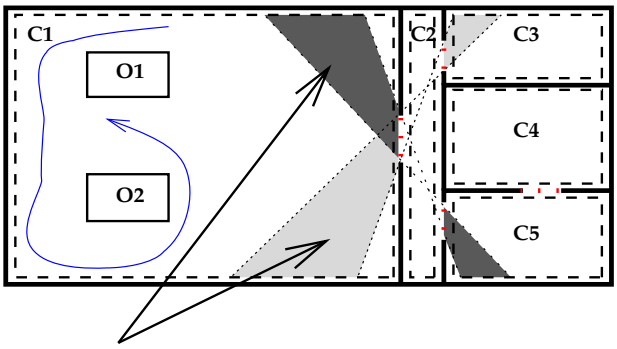

figure 19 : Cellule trop grande.

Dans cette figure, la cellule C1 comporte une ouverture à travers laquelle les cellules C3 et C5 sont visibles.

En effet, lorsqu'un observateur se déplace dans la cellule C1, et suit le trajet T, il ne voit pas les cellules C3 et C5. Or, pendant les calculs de visibilité, les polygones peuplant ces deux cellules ont tous été pris en compte.
La solution que nous proposons est un découpage de la cellule en deux volumes égaux, ainsi, le nombre de cellules prenant part aux calculs de visibilité est très largement réduit (figure 20).

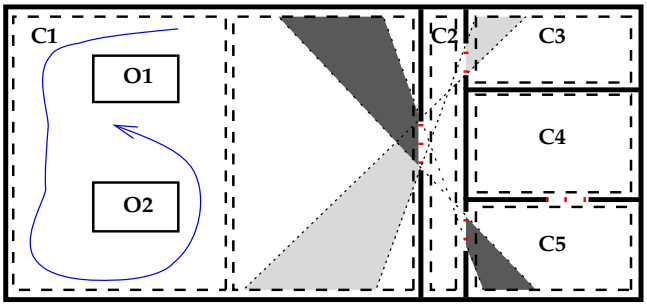

figure 20 : Découpage d'une cellule trop grande.

Seth Teller effectue un découpage de la cellule à traiter en tenant compte des ouverture [8].

3.5.2 Les cellules trop allongées

Malgré leur taille acceptable, certaines cellules peuvent êtres très allongées (figure 21). Nous retrouvons ce type de configuration dans des bâtiments contenant de très longs couloirs par exemple. Les cellules obtenues comportent généralement un grand nombre d'ouvertures. Ainsi, le nombre de facettes des PVS n'est quasiment pas réduit.

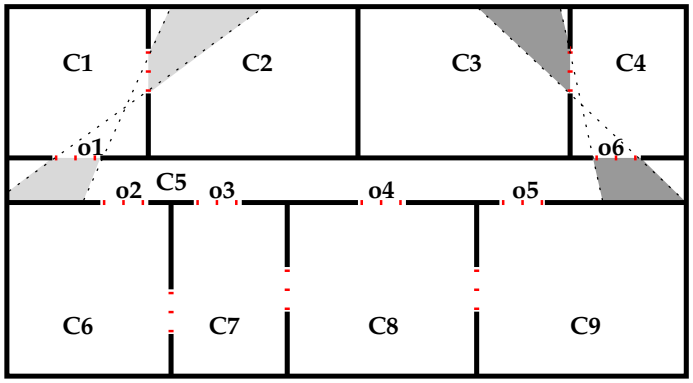

figure 21 : Cellule trop allongée.

Au cours d'un déplacement en temps réel (walkthrough) dans la cellule C5, les algorithmes de visualisation prennent en compte tous les éléments de la scène, quelque soit la position de l'observateur. De la cellule C5, toutes les cellules adjacentes sont visibles, ainsi que les cellules C2 et C3 par des séquences d'ouvertures.

Ce type de problème reste identique à celui cité précédemment. Nous proposons de découper la cellule en deux volumes égaux (figure 22).

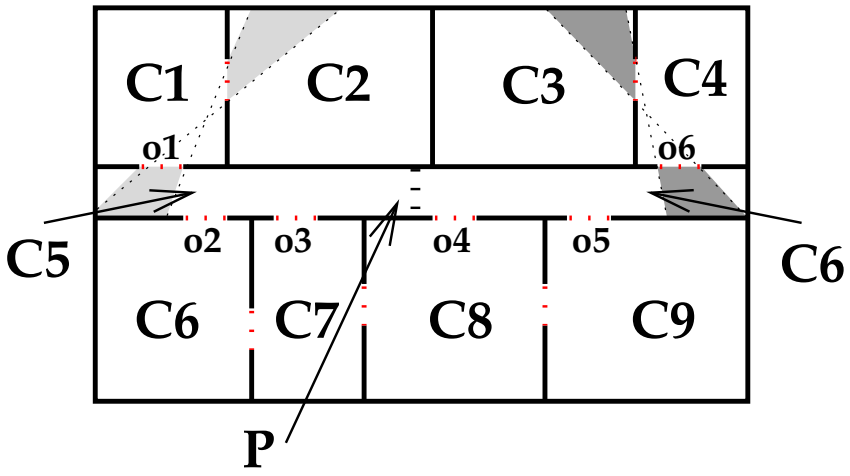

figure 22 : Découpage de cellule trop allongée.

Dans cette figure, la cellule C5 est subdivisée. Ainsi, le nombre de polygones potentiellement visibles de C5 est réduit.

3.5.3 Les fausses ouvertures

Ce problème survient lors du filtrage permettant d'obtenir les polygones potentiels de découpage. Une partie des murs de la scène ne sont pas candidats. Lorsque ces murs forment une concavité dans une cellule, des ouvertures pourtant inexistantes sont calculées (figure 23).

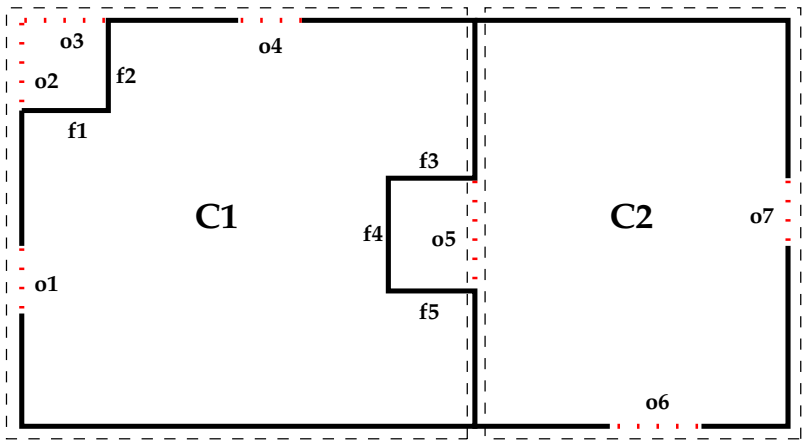

figure 23 : pièce non convexe.

Dans cette figure, les ouvertures O2, O3 et O5 sont de *fausses ouvertures*. En effet, contrairement aux ouvertures O1 et O4, elles ne permettent pas d'établir de relation de visibilité avec les cellules voisines.

Une solution consiste à détecter les plans à prendre en compte, puis à poursuivre la subdivision le long de ceux-ci (figure 24).

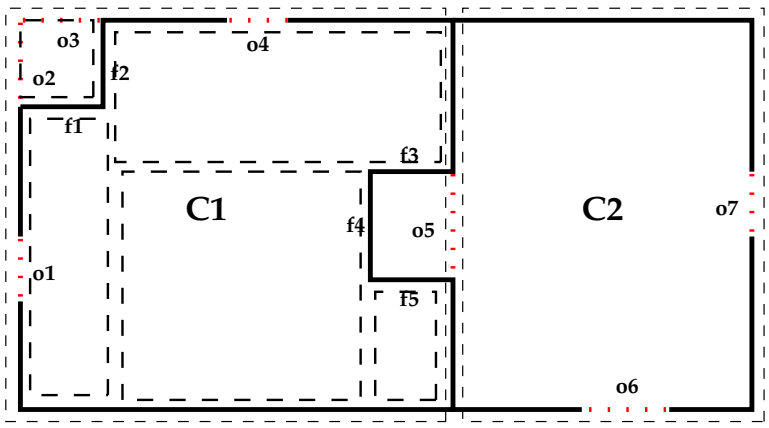

figure 24 : Découpage d'une pièce non convexe.

Dans cette figure, les plans détectés sont f1, f2, f3, f4, f5. Ils sont donc considérés comme des plans potentiels de découpage par notre algorithme. Le plan f2 est d'abord choisi, puis les plans f3 et f5 nous permettent ainsi d'obtenir une structuration correcte.

4 Etude des résultats

L'objectif des travaux proposés est de réduire la complexité des calculs de visibilité en structurant les scènes. Plusieurs paramètres permettent d'évaluer l'influence des critères de découpage retenus. Après avoir prsenté les scènes sur lesquelles les tests sont effectués, nous verrons comment interpréter les résultats obtenus.

4.1 Scènes traitées

Ces algorithmes ont été appliqués à plusieurs scènes. Nous en présentons deux des plus représentatives dans ce rapport.

4.1.1 Le loft

Cette scène est composée de deux pièces meublées. Elle comporte en tout 175 facettes dont 111 sont rectangulaires et axiales (figure 25).

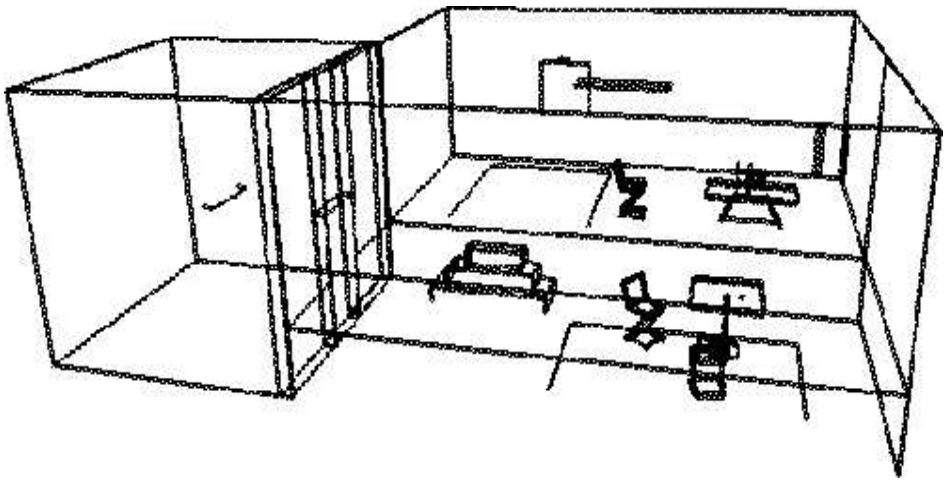

figure 25 : Le loft

4.1.2 Teller étage 1

L'étage 1 de Teller est beaucoup plus complexe. Il est composé d'une cinquantaine de pièces, de longs couloirs et de corridors. 2003 facettes composent cet étage. 1956 facettes sont rectangulaires et axiales (figure 26).

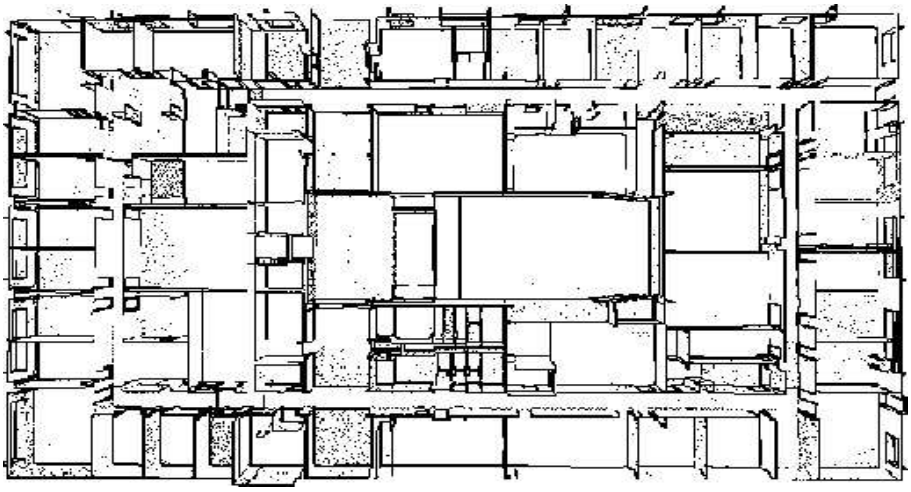

figure 26 : L'étage de S. Teller

4.2 Les paramètres mesurés

Les différents paramètres mesurés portent sur l'étape de structuration d'une part et sur la construction du graphe de visibilité d'autre part. Les premiers sont consacrés à l'influence des critères de découpage sur la structuration de la scène. Les seconds permettent d'évaluer l'efficacité de l'algorithme de construction du graphe de visibilité.

4.2.1 Statistiques de structuration

Les paramètres mesurés sont les suivants.

Pour le découpage de l'arbre.

- *"Nb de noeuds"* est le nombre de plans de découpage, ou le nombre de noeuds présents dans l'arbre,
- *"Prof maximum"* et *Prof. moyenne* sont respectivement les profondeurs maximum et moyenne de l'arbre obtenu.
- *"% faces sur plans"* est le pourcentage de facettes sur les plans de découpage.
  Les autres facettes, plus petites, ne font pas partie des éléments structurants majeurs de la scène.
- *"% faces sup."* est le pourcentage d'augmentation du nombre de facettes. Ce nombre est fonction du nombre de facettes en intersection avec les plans de découpage.
- *"Temps calcul BSP"* est le temps d'exécution pris pour établir l'arbre BSP.

Pour les cellules de l'arbre

- Le nombre de cellules obtenues *(Nb cellules)*.
- Le pourcentage de cellules avec facettes incluses, ou *cellules peuplées (% cell. peuplées)*.
- Le pourcentage de cellules avec *faces sur plans (% cell. murales)*.
  Les facettes murales sont les polygones se trouvant sur les frontières des cellules.
- Le pourcentage de cellules vides *(% cell. vides)*. Ces cellules ne contiennent aucune facette. Elles représentent l'intérieur d'un mur ou d'une colonne par exemple. Nous les appelons aussi fausses cellules.
- Le temps d'exécution pour calculer les facettes murales *(Temps calcul murs)*.

Pour les ouvertures entre cellules

- Le nombre d'ouvertures obtenues *(Nb ouvertures)*.
  Cette valeur indique en fait le nombre d'arcs dans le graphe d'adjacence.
- Le pourcentage d'ouvertures entre cellules vides *(% vides-vides)*.
- Le pourcentage d'ouvertures de cellules vides à cellules pleines *(% vides-pleines)*.
- Le pourcentage d'ouvertures entre cellules pleines *(% pleines-pleines)*.
- Le temps d'exécution pour calculer les ouvertures *(Temps calcul ouv.)*.

4.2.2 Calculs de visibilité.

Le graphe de visibilité représente les relations de visibilité existantes entre les cellules issues de l'étape de structuration. Ce calcul est effectué, à l'aide d'un lancer de rayons stochastique entre les différentes ouvertures. Les paramètres mesurés sont les suivants.

- Le nombre de rayons lancés *(Nb rayons lancés)*.
- Le nombre de tests de visibilité inter-ouvertures *(Nb test ouv.-ouv.)*.
- Le pourcentage de tests réussis *(% tests réussis)*.
- Le nombre de liens de visibilités produits *(Nb liens générés)*.
  Ce nombre correspond au nombre d'arcs dans le graphe de visibilité.
- Le temps d'exécution pour effectuer ces calculs *(Temps visibilité)*.

4.3 Résultats obtenus

Nous avons regroupé nos résultats dans une série de 4 tableaux pour chacune des scènes. Chacun d'entre eux présente les résultats des points proposés dans la partie précédente.

Les colonnes *opacité* représentent les chiffres obtenus par le filtrage préliminaire. Les sous-colonnes C1, C2 et C3 ont les significations suivantes.

C1
Cette colonne représente les chiffres obtenus grâce au critère de découpage par plan médian.

C2
Le second critère est celui du plan ayant l'opacité maximale.

C3
Ce dernier critère est celui dont le plan choisi découpe un minimum de polygones.

4.3.1 Le loft

Voici les résultats obtenus sur le loft. Le découpage BSP du loft présente les propriétés suivantes (figure 27).

	Opacité = 0			**Opacité = 0.3**			**Opacité = 0.7**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb de noeuds	499	227	185	45	31	29	33	31	31
Nb de feuilles	250	114	93	23	16	15	17	16	16
Prof. maximum	13	26	28	9	12	12	11	12	11
Prof. moyenne	9	15	16	6	7	7	7	7	7
% faces supp.	196.6	148	121.1	6.8	6.8	6.8	6.8	6.8	6.8
% faces sur plans	73.2	69.6	64.6	18.7	18.7	18.7	18.7	18.7	18.7
Temps calcul BSP	< 1 s.								

figure 27 : Statistiques du découpage BSP du loft.

Pour un critère de préfiltrage où l'opacité est fixée à 0, l'algorithme choisit tous les plans comme candidats potentiels au découpage. Nous sommes donc en présence cette fois du cas où de petits polygones peuvent êtres choisis comme plans de découpage.

Bien entendu, dans ce cas, la topologie de la scène n'est pas repectée. En d'autres termes, les cellules obtenues seront en inadéquation avec les pièces de la scène.

Le tableau comportant les statistiques sur les cellules obtenues (figure 28) représente bien cet état.

	Opacité = 0			**Opacité = 0.3**			**Opacité = 0.7**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb cellules	250	114	93	23	16	15	17	16	16
% cell. vides	43.2	12.3	12.9	69.6	56.2	53.3	58.8	56.2	56.2
% cell. murales	55.6	85.9	87.1	30.4	43.8	46.7	41.2	43.8	43.8
% cell. peuplées	10.4	18.4	20.4	13.0	18.8	20	17.6	18.8	18.8
Temps calcul murs	< 1 s.								

figure 28 : Statistiques des cellules du loft.

Dans les scènes architecturales, la prise en compte de l'épaisseur des murs provoque la création de cellules *vides*. Ces cellules sont à l'intérieur des murs et ne contiennent évidemment aucun objet.

Le loft est composé de trois pièces, or le nombre de cellules s'élève à 93 dans le meilleur des cas pour un préfiltrage d'opacité minimale à 0.0.
Le nombre de cellules est trop important, il en résulte une mauvaise structuration. Les calculs d'ouvertures sont donc inappropriés, et les graphes de visibilité inadaptés. (figure 29).
En revanche, dès que le seuil du degré d'opacité augmente, la réduction du nombre de cellules est importante. Cela prouve bien l'importance du préfiltrage dans la structuration d'une scène.

	Opacité = 0			**Opacité = 0.3**			**Opacité = 0.7**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb ouvertures	754	383	293	43	28	26	29	26	26
% vides-vides	35.9	5.7	7.8	86	78.6	76.9	79.3	76.9	76.9
% pleines-pleines	62.9	91.9	90.1	14	21.4	23.1	20.7	23.1	23.1
% pleines-vides	1.2	2.3	2.0	0	0	0	0	0	0
Temps calcul ouv.	< 1 s.								

figure 29 : statistiques sur les ouvertures du loft.

Lorsque le degré d'opacité du préfiltrage est non nul, le nombre de rayons lancés est sensiblement identique (figure 30). Ce tableau prouve, encore une fois l'utilité du préfiltrage par critère d'opacité.

	Opacité = 0			**Opacité = 0.3**			**Opacité = 0.7**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb rayons lancés	13175888	6179751	3534926	58	58	55	64	61	59
Nb tests ouv.-ouv.	1475023	697720	403838	18	18	18	18	18	18
% tests réussis	15.4	16	18.1	77.8	77.8	77.8	77.8	77.8	77.8
Nb liens générés	12452	5934	4012	40	40	40	40	40	40
Temps visibilité	371 s.	177 s.	97 s.	< 1 s.					

figure 30 : statistiques sur la visibilité du loft.

En revanche, lorsque le degré d'opacité est nul, le nombre de cellules est très important. Ainsi, une grande quantité de rayons sont nécessaire à l'évaluation du graphe de visibilité et les temps de calculs sont très élevés.

En prenant comme valeurs 0.3 pour le préfiltrage et le critère d'opacité maximum, nous obtenons une structuration satisfaisante du loft (figure 31).

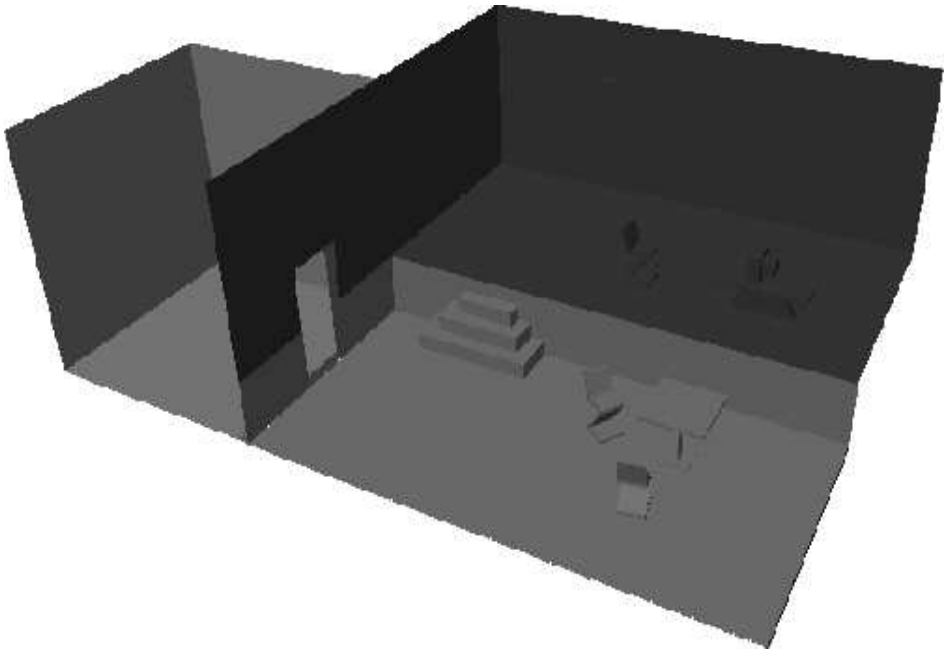

figure 31 : Structuration du loft.

4.3.2 L'étage de S. Teller.

Sur une scène telle que celle de l'étage de S. Teller, le découpage BSP joue un rôle prépondérant. En effet, le nombre de polygones étant beaucoup plus élevé la

structuration de la scène doit être satisfaisante (figure 32).

	Opacité = 0			**Opacité = 0.4**			**Opacité = 0.6**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb de noeuds	5793	2367	2071	2425	2035	1419	11	11	11
Nb de feuilles	2897	1184	1036	1213	1018	710	6	6	6
Prof. maximum	17	39	46	28	39	34	6	6	6
Prof. moyenne	13	24	26	16	24	22	4	4	4
% faces supp.	116.8	42.9	25.8	46.3	40.4	22.1	1.2	1.2	1.2
% faces sur plans	97.0	96.4	96.9	90.7	89.7	78.7	19.0	19.0	19.0
Temps calcul BSP	2 s.	2 s.	3 s.	1 s.	1 s.	2 s.	< 1 s.		

figure 32 : Statistiques du découpage BSP de l'étage.

Ce tableau nous montre combien le préfiltrage reste important dans les cas de scènes complexes.

De plus, la structuration de la scène est très dépendante de la valeur du seuil d'opacité. Entre 0.4 et 0.6, le seuil choisi entraine une forte variation des cellules obtenues.

Le critère C1 nous permet d'obtenir un arbre BSP équilibré. Cependant, la topologie de la scène n'est pas prise en compte, si ce n'est par le préfiltrage.

Le critère C3 nous permet réduire de manière importante le nombre de facettes découpées. Ainsi, l'arbre BSP est moins profond et le nombre de cellules est plus faible.

En modifiant la valeur du seuil de préfiltrage, nous obtenons une structuration très satisfaisante.

	Opacité = 0			**Opacité = 0.4**			**Opacité = 0.6**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb cellules	2897	1184	1036	1213	1018	710	6	6	6
% cell. vides	58.4	55.3	58.4	52.6	52.7	54.2	0	0	0
% cell. murales	41.4	44.4	41.4	47.2	46.9	45.2	83.3	83.3	83.3
% cell. peuplées	1.5	2.11	1.35	2.7	3.14	2.7	33.3	33.3	33.3
Temps calcul murs	< 1 s.								

figure 33 : Statistiques sur les cellules de l'étage.

Le second critère nous permet de prendre en compte la topologie de la scène. Le découpage est effectué selon les principaux éléments. Ainsi, les grandes facettes ne sont jamais découpées par de petits polygones.

Le nombre de facettes aux frontières des cellules est représentatif du respect de la topologie de la scène(figure 34).

	Opacité = 0			**Opacité = 0.4**			**Opacité = 0.6**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb ouvertures	8002	3413	3123	3448	2824	2270	1345	1345	1345
% vides-vides	62.1	67.4	69.8	47.9	36.7	36.2	0	0	0
% pleines-pleines	29.2	21.2	17.9	29.2	29.8	32.8	100	100	100
% pleines-vides	8.7	11.4	12.3	22.9	33.5	31.0	0	0	0
Temps calcul ouv.	27 s.	5 s.	5 s.	5 s.	3 s.	3 s.	12 s.	12 s.	12 s.

figure 34 : Statistiques des ouvertures de l'étage.

Les visibilités sont représentées par une séquence d'ouvertures. Plus le nombre de cellules est important, plus le graphe de visibilité est complexe, plus les séquences sont longues (figure 35).

	Opacité = 0			**Opacité = 0.4**			**Opacité = 0.6**		
	C1	C2	C3	C1	C2	C3	C1	C2	C3
Nb rayons ($\times 10^6$)	41.7	12.5	13.8	39.8	13.5	28.4	4.3	3.9	2.6
Nb tests ($\times 10^5$)	48.5	13.7	15.3	42.5	14.8	30.3	5.6	5.2	3.8
% réussis	19.8	11.9	13.3	8.8	11.2	8.6	26.4	27.5	36.7
Nb liens	94564	21478	14684	33032	18772	12118	30	30	30
Temps (en s.)	4738	1235	1468	3237	1061	2363	309	312	191

figure 35 : Statistiques de visibilité de l'étage.

Dans cet algorithme, le nombre de rayons lancés par ouverture s'élève à 10. Lorsqu'un rayon lancé d'une ouverture atteint la cellule cible, les tests sont interrompus car les deux cellules sont visibles. Le grand nombre de rayons lancés s'explique par le parcours en profondeur du graphe d'adjacence. Toutes les séquences d'ouvertures sont testées et seulement un petit nombre d'entre-elles sont correctes.

4.4 Les résultats obtenus par la méthode proposée par Airey

J. Airey propose une méthode permettant de prendre en compte parallèlement les trois critères de subdivision en leur affectant un coefficient de pondération. Le coefficient le plus fort est affecté au critère qu'il estime être le plus important. De cette manière, il ne donne pas la priorité absolue à un seul critère de subdivision.

les coefficients qu'il propose sont les suivants :

- 0.5 pour le plan le plus occlusif.
- 0.3 pour le plan permettant le meilleur équilibrage de l'arbre.
- 0.2 pour le plan découpant le moins de facettes.

Ce choix est issu des différents essais qu'il a effectué [22].

Nous avons regroupé les 4 tableaux précédents en un seul (figure 36). Notons que les temps de calcul de la construction de l'arbre BSP impliqués par la méthode d'Airey sont plus longs que ceux obtenus avec un seul critère de subdivision.

	Opacité = 0	**Opacité = 0.4**	**Opacité = 0.6**
Nb de noeuds	3055	2171	11
Nb de feuilles	1528	1086	6
Prof. maximum	34	31	6
Prof. moyenne	20	19	4
% faces supp.	57.8632	40.88	1.2
% faces sur plans	96.7426	90.4	19
Temps calcul BSP	5 s.	5 s.	5 s.
Nb cellules	1528	1086	6
% cell. vides	57.5262	52.2	0
% cell. murales	42.2775	47.3	83.3
% cell. peuplées	1.63613	2.57	33.3
Nb ouvertures	4256	3068	1345
% vides-vides	23	38.75	0
% pleines-pleines	65	30.63	100
% pleines-vides	10	30.60	0
Temps calcul ouv.	12 s.	12 s.	12 s.
Nb rayons ($\times 10^6$)	15.7	19.6	5.9
Nb tests ($\times 10^5$)	17.7	21.4	7.3
% réussis	16	11.5	20.2
Nb liens	28174	24592	30
Temps (en s.)	310	312	296

figure 36 : Statistiques du découpage BSP de l'étage, méthode Airey.

Le choix du seuil d'opacité influe de manière significative sur le découpage et les calculs de visibilité. En effet, pour un seuil nul la structuration ne répond pas à nos objectifs puisque tous les polygones sont pris en compte. Pour un seuil de 0.6 toutes les valeurs du tableau ci-dessus sont identiques, quelque soit la méthode utilisée. Pour un seuil de 0.4, la méthode proposée par J. Airey est plus efficace et permet un gain de temps considérable sur les calculs de visibilité.

4.5 Analyse et discussion

Les algorithmes présentés ci-dessus permettent d'obtenir une structuration de la scène en cellules. Pour effectuer cette structuration, la mise en oeuvre de l'algorithme s'effectue en deux phases :

- *un prétraitement sur un critère d'opacité*
  Cette phase permet de filtrer les objets pour ne garder que les plus occlusifs (sols, murs, etc.).

- *la subdivision proprement dite*
  Cette étape structure la scène en un arbre BSP à l'aide de trois critères : occlusion, plan médian et découpage minimal. Airey propose de combiner ces trois critères en leur affectant un poids.

Le prétraitement fournit des résultats significatifs seulement pour des seuils d'opacité suffisants. En effet, pour des seuils très faibles, tous les polygones sont pris en compte lors du découpage, y compris ceux de très petite surface.

Dans la seconde phase, les critères d'opacité et de découpage minimal fournissent les meilleurs résultats en terme de cellules proches des pièces de la scène.

4.6 Interprétation des coefficients de la méthode Airey

Le choix des coefficients de pondération peut s'interpréter de la façon suivante. Soit $\mathcal{A}$ le plan le plus occlusif. Supposons que l'arbre BSP construit selon $\mathcal{A}$ soit déséquilibré et que de nombreuses facettes soient découpées. Supposons aussi qu'il existe un plan $\mathcal{B}$ légèrement moins occlusif que $\mathcal{A}$ permettant d'obtenir un arbre BSP bien équilibré tout en ne découpant que peu de facettes. Alors ce plan $\mathcal{B}$ permet une meilleure structuration de la scène.

Si la priorité absolue est donnée au plan le plus occlusif, la structuration de la scène ne répond pas aux objectifs fixés. En revanche, la méthode proposée par Airey permet de choisir effectivement le plan $\mathcal{B}$ au lieu du plan $\mathcal{A}$ car les trois critères de subdivision sont pris en compte.

Le choix des poids proposés par J. Airey est empirique. Cependant, Grâce à l'étude statistique que nous avons menée, ce choix est justifiable :

- le plan le plus occlusif permet d'obtenir une structuration très proche des pièces de la scène. Puisque c'est notre objectif principal, le coefficient de pondération associé à cette propriété doit donc être plus important.

- Le second critère favorise la construction d'un arbre bien équilibré et permet ainsi d'accélérer l'algorithme de construction de l'arbre BSP.
- le dernier critère, de moindre importance selon Airey, permet de ne pas trop augmenter la taille de la base de données.

Les poids sont affectés en fonction de la priorité que l'on donne aux critères de subdivision. Ainsi, ils peuvent être modifiés en fonction du résultat que l'on souhaite obtenir. Notre principal objectif est de structurer la scène en respectant au mieux sa topologie, c'est pourquoi nous avons, comme Airey, favorisé le choix du plan le plus occlusif.

5 Conclusion

Dans cette étude, nous nous sommes intéressés à diminuer la complexité du calcul de visibilité sur des scènes axiales. Ces travaux intéressent, en particulier, le rendu photo-réaliste de scènes complexes ainsi que leur visualisation interactive.

Notre approche consiste à précalculer certaines relations de visibilité entre les objets d'une scène. Pour cela, nous proposons deux étapes principales: structurer la scène en cellules et ouvertures, puis calculer un graphe de visibilité entre ces cellules. L'étape de structuration a été particulièrement approfondie pour examiner les différents paramètres conduisant rapidement à un graphe de visibilité performant.

Pour structurer la scène, nous utilisons un découpage de la scène à l'aide de plans de partition binaire. Le critère d'opacité, déjà introduit par Airey, permet ainsi de retenir plusieurs plans de découpage. Cependant, ce critère reste insuffisant pour obtenir une structuration efficace de la scène. Nous avons donc rajouté trois critères secondaires différents afin d'obtenir une meilleure structuration.

Les résultats obtenus par notre algorithme montrent que le critère d'opacité, couplé au critère de réduction du nombre de coupures, donne une bonne structuration de la scène: les cellules obtenues suivent de près les pièces et couloirs existant dans la scène.

Dans un souci d'améliorer les performances de notre algorithme de structuration, nous avons également apporté des solutions aux problèmes posés par les cellules trop grandes, trop allongées ou encore non convexes.

En tirant parti de cette structuration, nous construisons le graphe de visibilité entre les cellules. Ce graphe indique pour chaque cellule les cellules qui lui sont visibles. Les différents fichiers générés par notre algorithme permettent de représenter facilement ce graphe.

En utilisant ce graphe de visibilité pour traiter des modèles géométriques complexes, il est alors possible d'accélérer les calculs d'illumination globale, en ne chargeant en mémoire que les cellules participant aux échanges lumineux. De même, ce graphe de permet de réduire les calculs de visibilité lorsqu'un utilisateur souhaite se promener dans une scène complexe de manière interactive. Ainsi, seules les cellules visibles de la cellule occupée par l'observateur peuvent être chargées en mémoire pour visualiser la scène.

Malgré cela, plusieurs problèmes restent à résoudre. En effet, l'algorithme calculant le graphe de visibilité est peu efficace dès que le nombre de cellules augmente. La gestion du graphe de visibilité reste également à étudier pour optimiser les échanges entre la mémoire et la base de données regroupant les cellules d'une scène. Une parallélisation de cette gestion du graphe permettrait certainement de réduire ces coûts de communication.

Enfin, le passage aux scènes non axiales permettrait de traiter des scènes plus générales.

Références

[1] K. Bouatouch & M. Madani & T. Priol & B. Arnaldi.
A New Algorithm of Space Tracing Using a CSG Model.
Eurographics'88, 1988.

[2] Glassner, A. S.
Space subdivision for Fast Ray Tracing.
IEEE Computer Graphics and Applications, 1984.

[3] Jean-Philippe Thirion.
Tries: Data structures based on binary representation for ray tracing.
Eurographics '90, C.E. Vandoni and D.A. Duce, Eurographics Association, Elsevier Science Publishers, 1990

[4] C. B. Jones.
A new approach to the hidden line problem.
The Computer Journal, March 1971.

[5] Frederick P. Brooks, John M. Airey, John H. Rohlf.
Towards Image Realism with Interactive Update Rates in Complex Virtual Building Environments.
ACM, May 1990.

[6] Harry Plantiga & Charles R. Dyer.
Visibility, Occlusion, and the Aspect Graph.
International Journal of Computer Vision, 1990.

[7] Seth J. Teller & Carlo H. Sequin.
Visibility Preprocessing For Interactive Walkthroughs.
Computer Graphics, July 1991.

[8] Seth J. Teller.
Visibility Computations in Density Occluded Polyhedral Environments.

Phd Thesis, University of California at Berkeley, 1992.

[9] Seth Teller & Pat Hanrahan.
Global Visibility Algorithms for Illumination Computations.
Computer Graphics Proceedings, Annual Conference Series, 1993.

[10] Seth Teller & Celeste Fowler & Thomas Funkhouser & Pat Hanrahan.
Partitioning and Ordering Large Radiosity Computations.
Computer Graphics Proceedings, Annual Conference Series, 1994.

[11] Pat Hanrahan & David Salzman & Larry Aupperle.
A Rapid Hierarchical Radiosity Algorithm.
Computer Graphics, July 1991.

[12] David Luebke, Chris Georges.
Portals and Mirrors: Simple, Fast Evaluation of Potentially Visible Sets.
Symposium on Interactive 3D Graphics, 1995.

[13] M. Gilliland, R. A. Schumacker, R. Brand, & W. Sharp.
Study for Applying Computer-Generated Images to Visual Simulation.
U.S. Air Force Human Resources Laboratory, 1969.

[14] H. Fuchs & Z. Kedem & B. Naylor.
On visible surface generation by a priori tree structures.
Computer Graphics, March 1980.

[15] Bruce Naylor, John Amanatides, William Thibault.
Merging BSP Trees Yields Polyhedral Set Operations.
Computer Graphics, Aug 1990.

[16] Norman Chin & Steven Feiner.
Near Real-Time Shadow Generation Using BSP Trees.

Computer Graphics, July 1989.

[17] Michael S. Paterson & F. Frances Yao.
Efficient Binary Space Partitions for Hidden-Surface Removal and Solid Modeling.
Discrete and Computational Geometry, May 1990.

[18] Enric Torres.
Optimization of the binary space partition algorithm (BSP) for the visualisation of dynamic scenes.
Eurographics '90, C. E. Vandoni and D. A. Duce, Eurographics Association, 1990.

[18] Bruce F. Naylor.
Partitioning Tree Image Representation and Generation from 3D Geometric Models.
Graphics Interface '92, 1992.

[21] Paz Morer, Alejandro M. Garcia-Alonso, & Juan Flaquer.
Optimization of a priority list algorithm for 3-D rendering of buildings.
Computer Graphics forum, Vol.14 1995.

[22] John M. Airey.
Increasing update rates in the building walkthrough system with automatic model-space subdivision and potentially visible set calculation.
PhD Thesis, July 1990.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399